Software Studio
軟體設計與實驗

Animation Tutorial

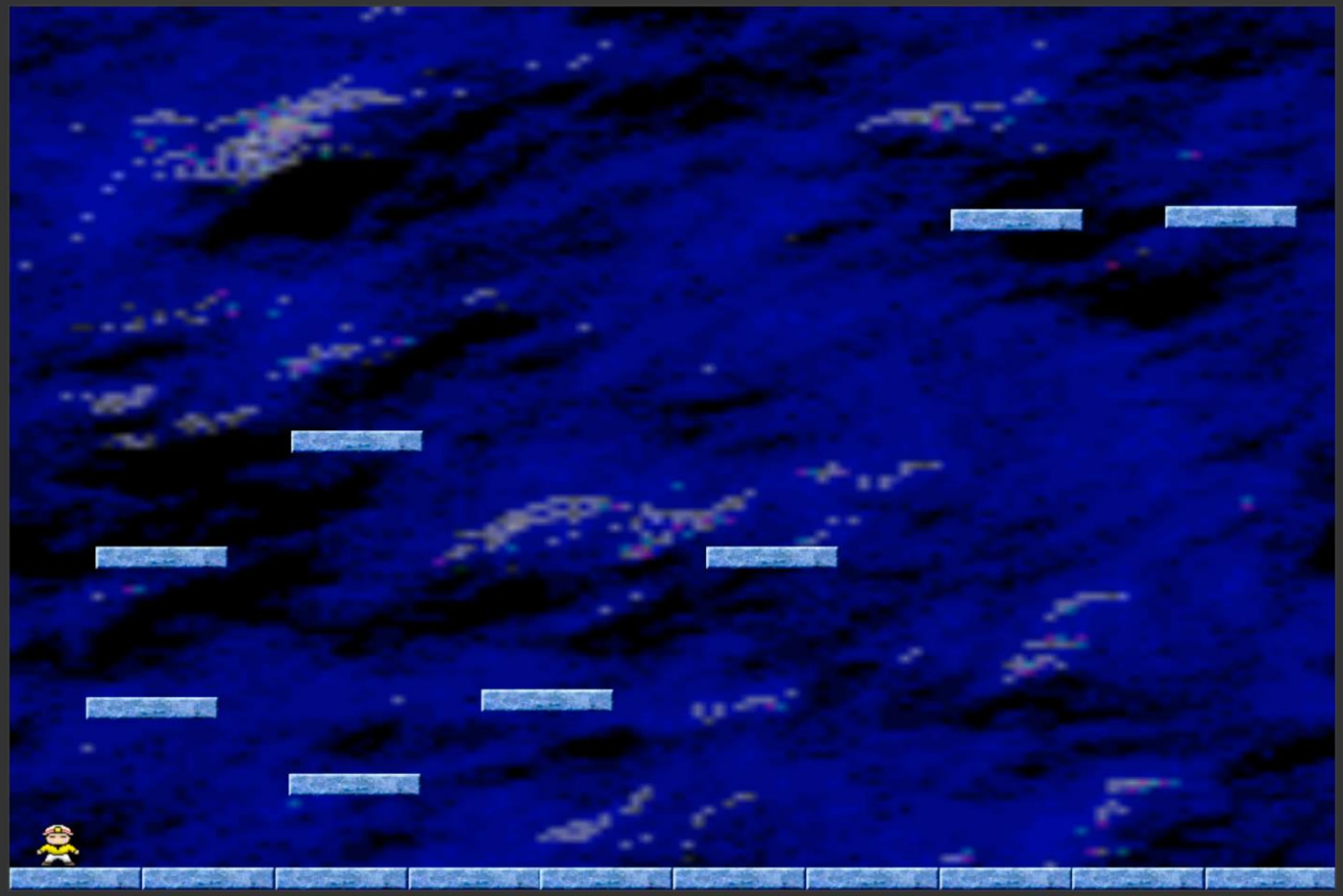
Hung-Kuo Chu

Department of Computer Science

National Tsing Hua University

CS2410

Goal

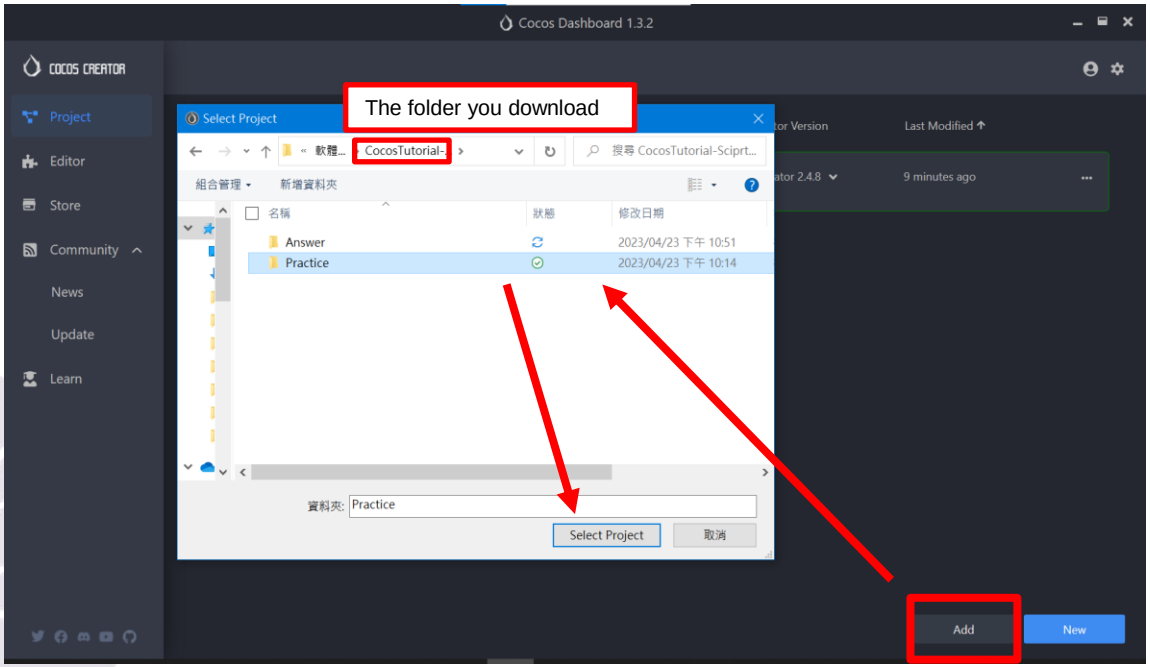


Contents

- Create walk animation
- Create fall animation
- Create moving platforms animation

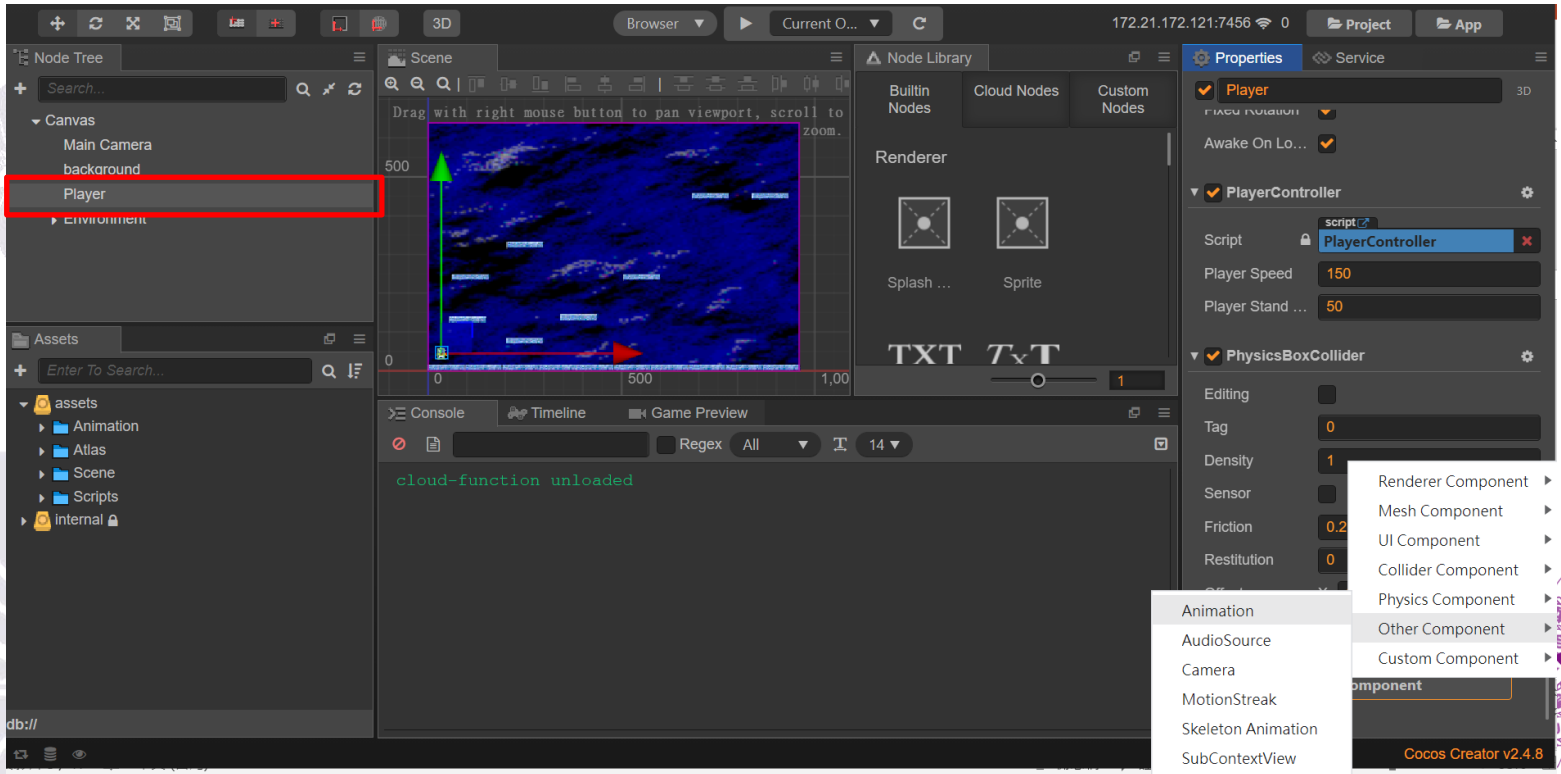
Open the project

- Step 1: Download project from eeclass or GoogleDrive and unzip
  - https://reurl.cc/V8yAry
- Step 2: Add the Practice folder to Cocos



Create walk animation

- Step 1: Create animation component
  - Choose the Player node
  - Add component → Other component /Animation



Create walk animation

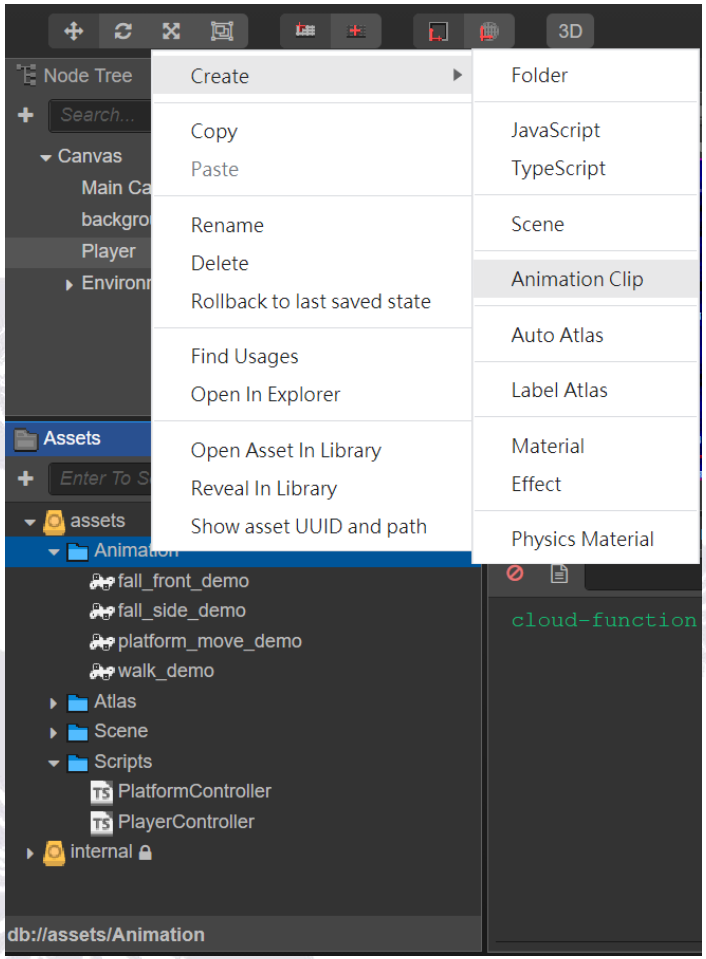
- Step 2: Get animation component in PlayerController.ts

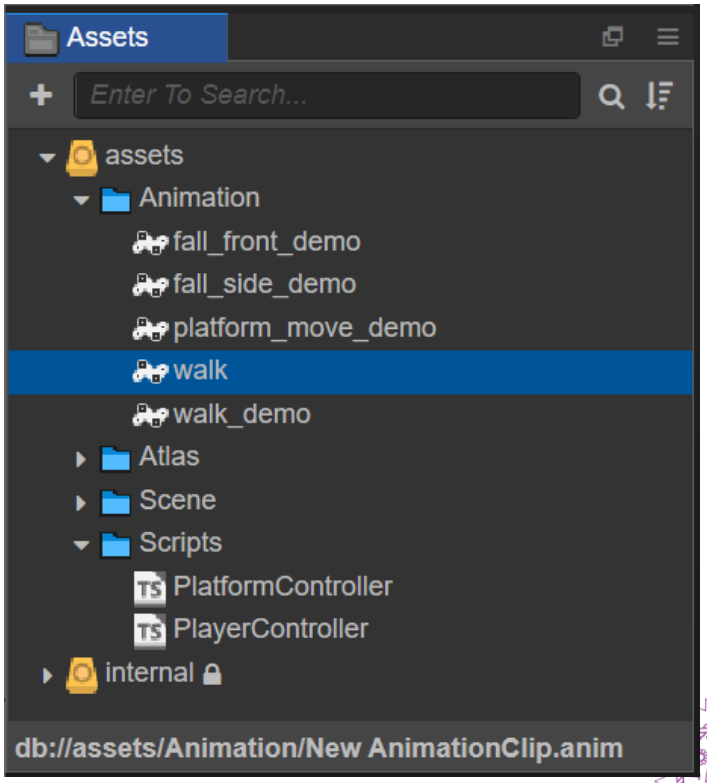
assets > Scripts > TS PlayerController.ts > PlayerController > start

```
36      start (){
37          this.idleFrame = this.getComponent(cc.Sprite).spriteFrame;
38          // ========== TODO ==========
39          // 1. Assign the animation component into this.anim from this.node
40          this.anim = this.getComponent(cc.Animation);
41      }
```

Create walk animation

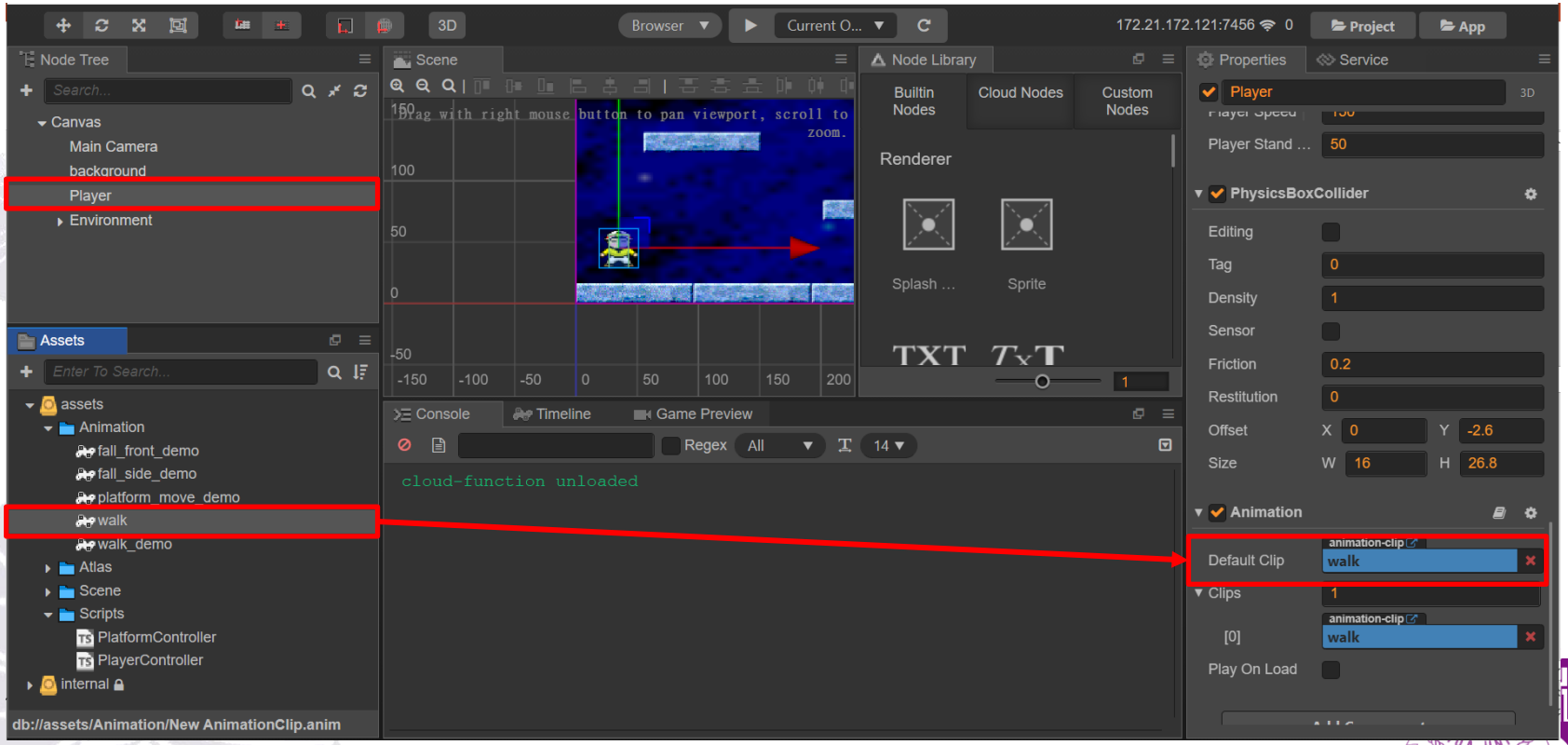
- Step 3: Create an animation clip, named “walk”





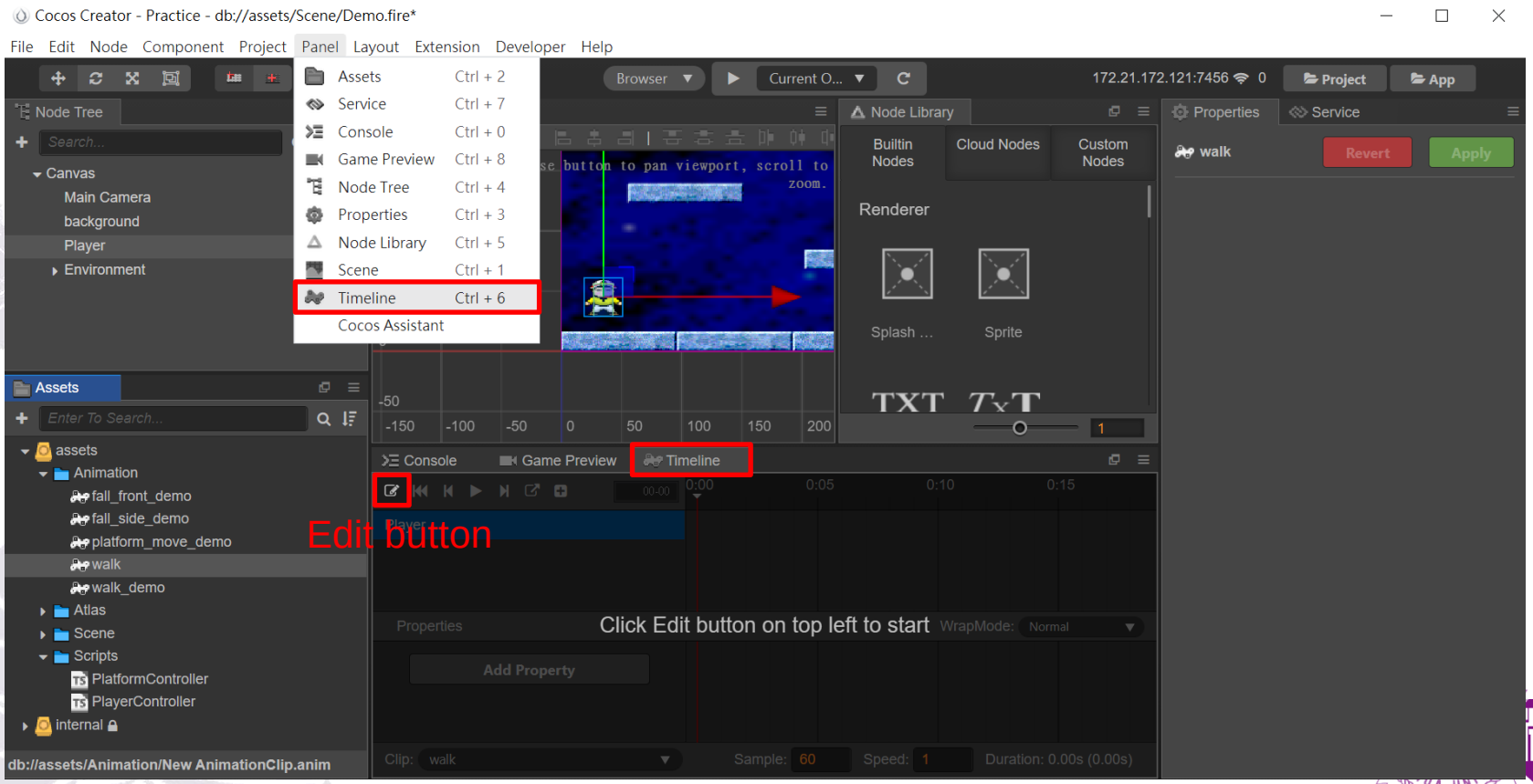
Create walk animation

- Step 4: Put the “walk” animation clip to the animation component



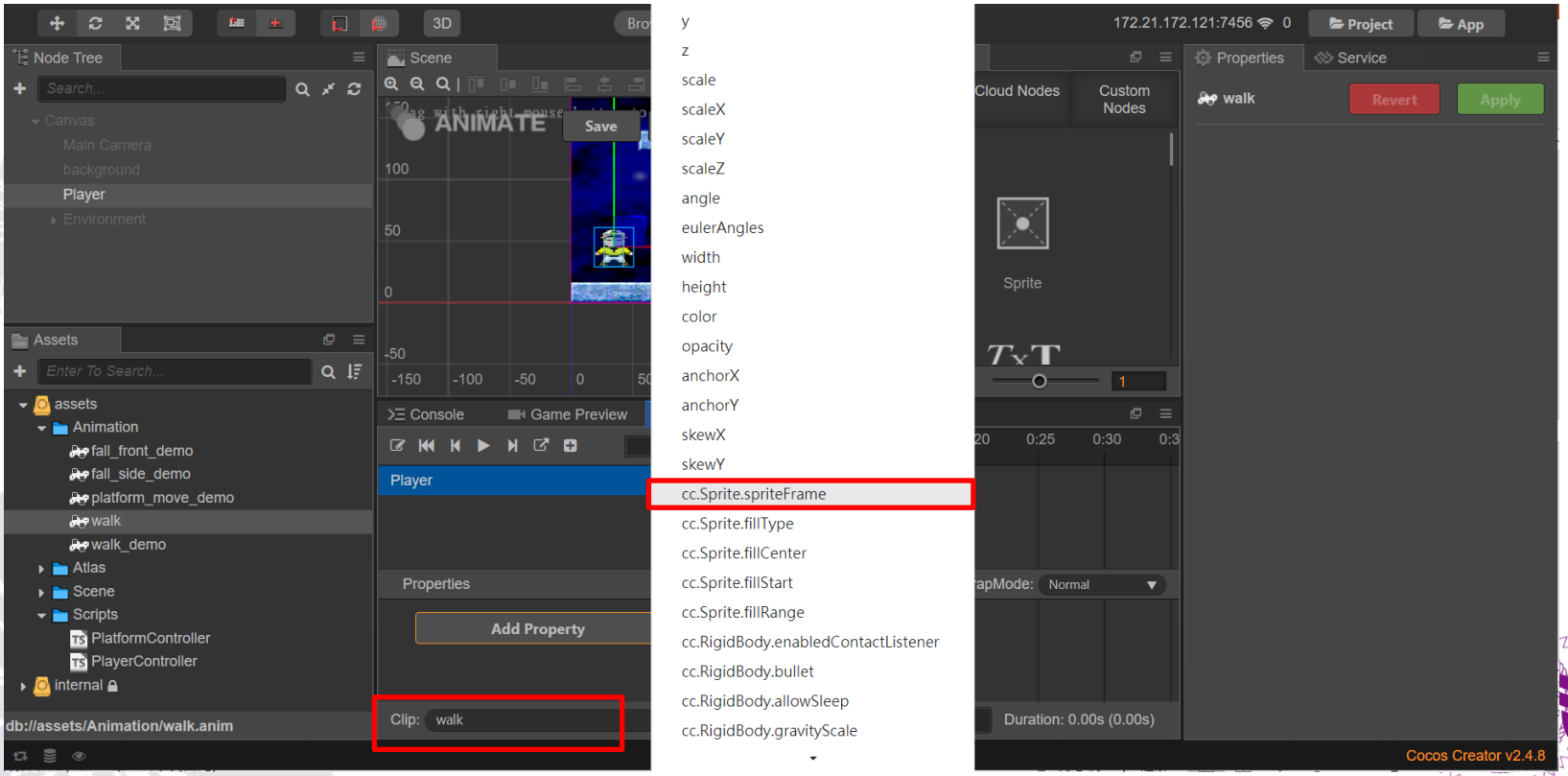
Create walk animation

- Step 5: Open animation editor (Timeline)



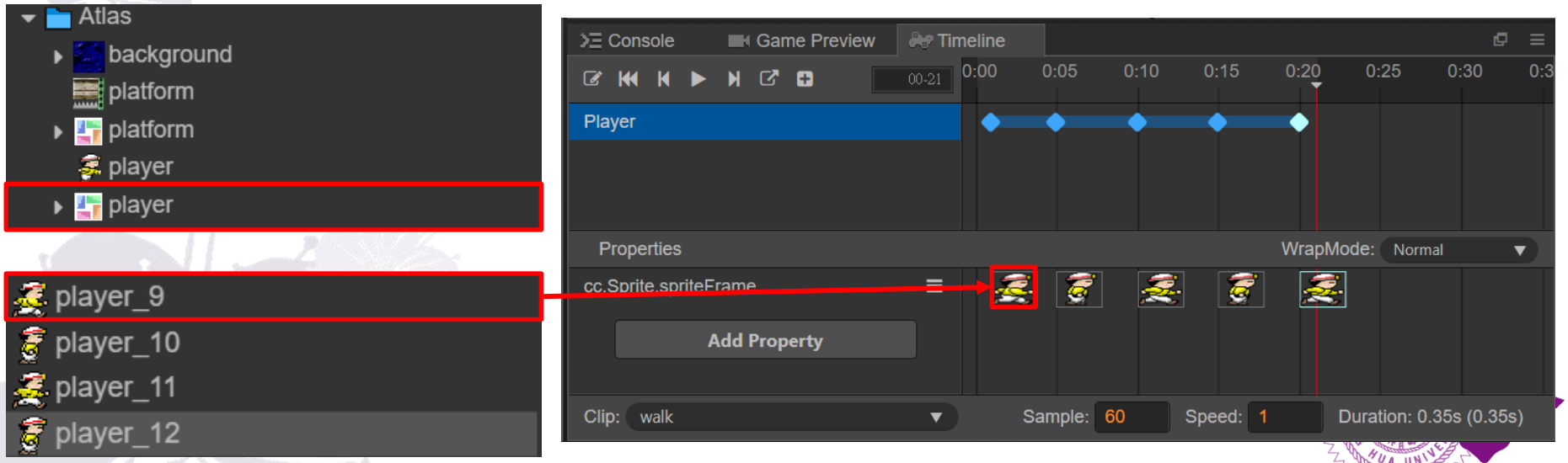
Create walk animation

- Step 6: Choose the “walk” animation clip
- Step 7: Add Property→cc.Sprite.spriteFrame



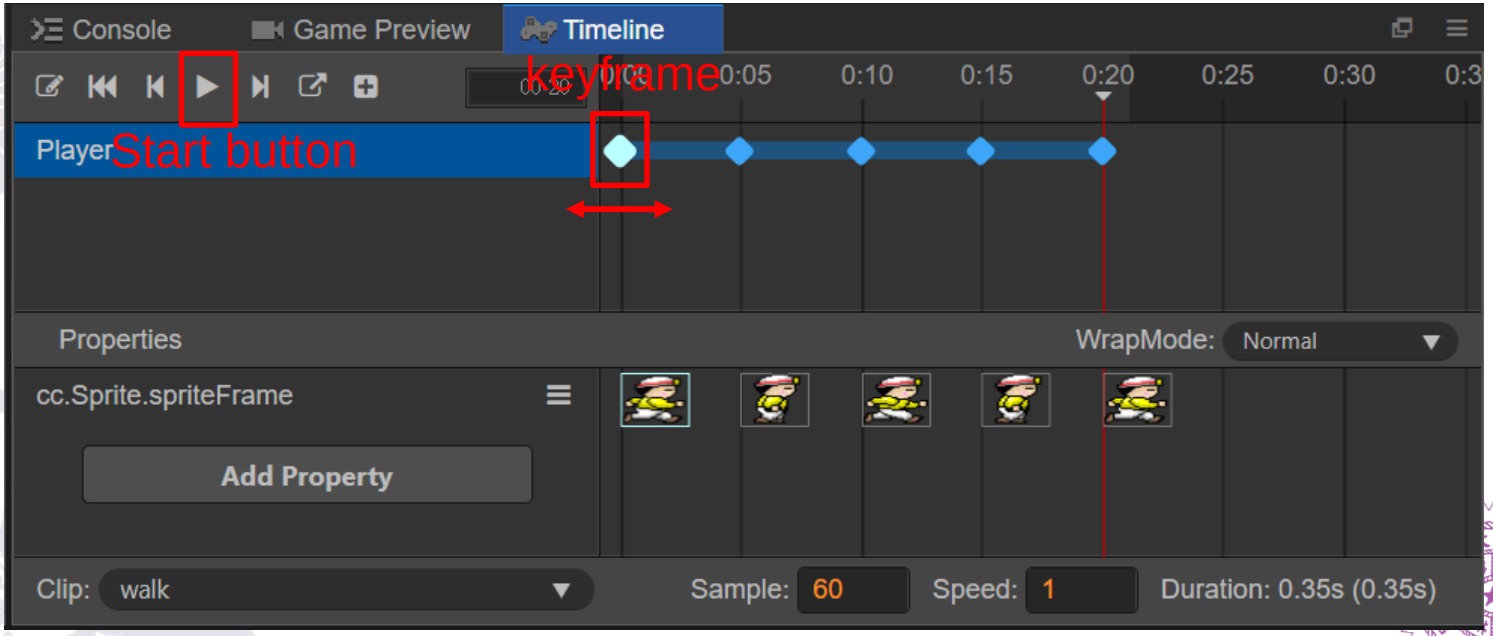
Create walk animation

- Step 8: Drag the sprite to animation editor
  - Sprite location: Atlas/player
  - Sprite order: player_9→ player_ 10→ player_ 11→ player_ 12→ player_ 9



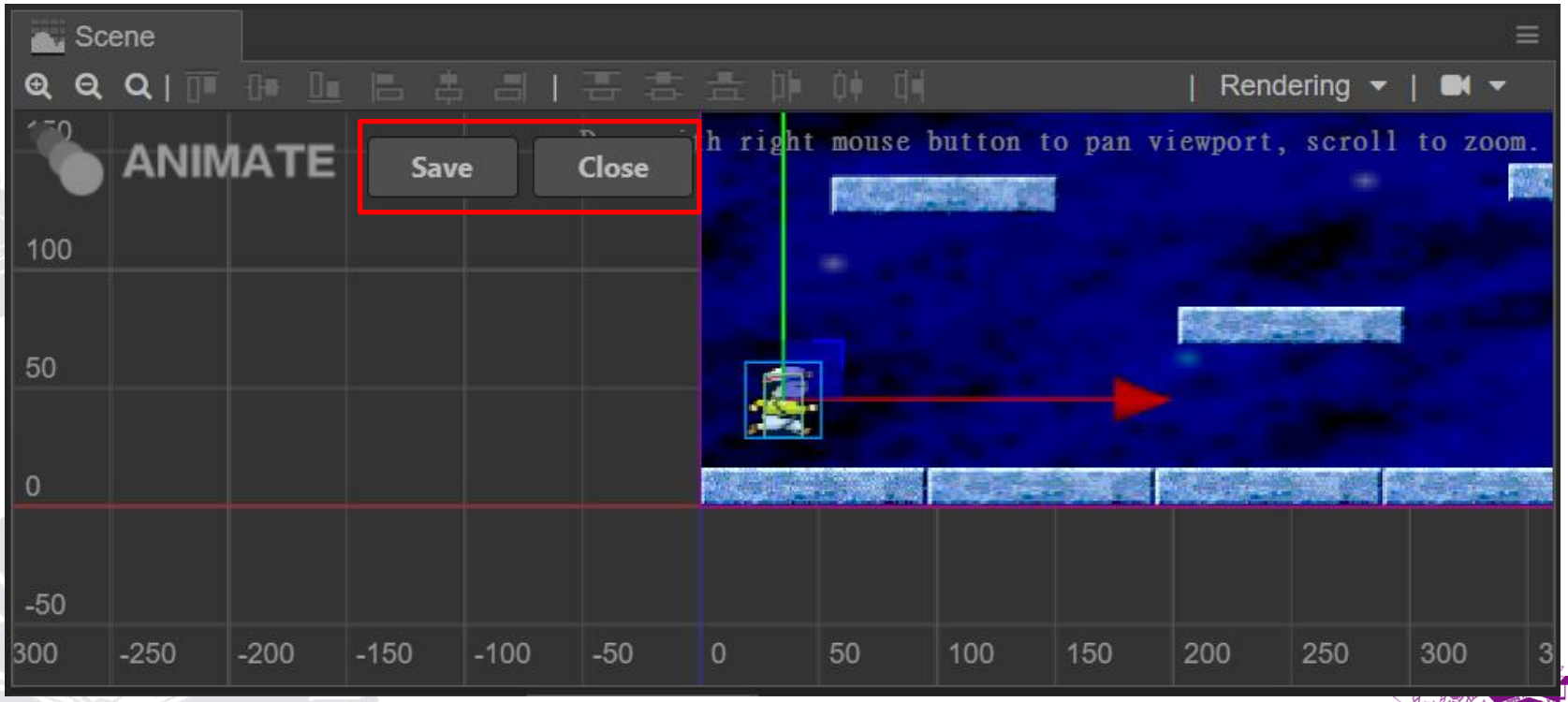
Create walk animation

- Step 9: Adjust the time interval between each sprite
  - Drag keyframe directly
  - Press the start button to preview animation



Create walk animation

- Step 10: Save the changes and close the editing status



Create walk animation

- Step 11: Play the animation with the script
- Step 12: Stop the animation if the player is not moving

assets > Scripts > TS PlayerController.ts > PlayerController > playerAnimation

```
107             if(this.moveDir == 0)
108             {
109                 this.getComponent(cc.Sprite).spriteFrame = this.idleFrame;
110                 // ========== TODO ==========
111                 // 1. Stop the animation which is playing
112                 this.anim.stop();
113             }
114
115             // ========== TODO ==========
116             // 1. Play walk animation (Checked the walk animation is playing or not)
117             else if(!this.anim.getAnimationState("walk").isPlaying){
118                 this.anim.play("walk");
119             }
```

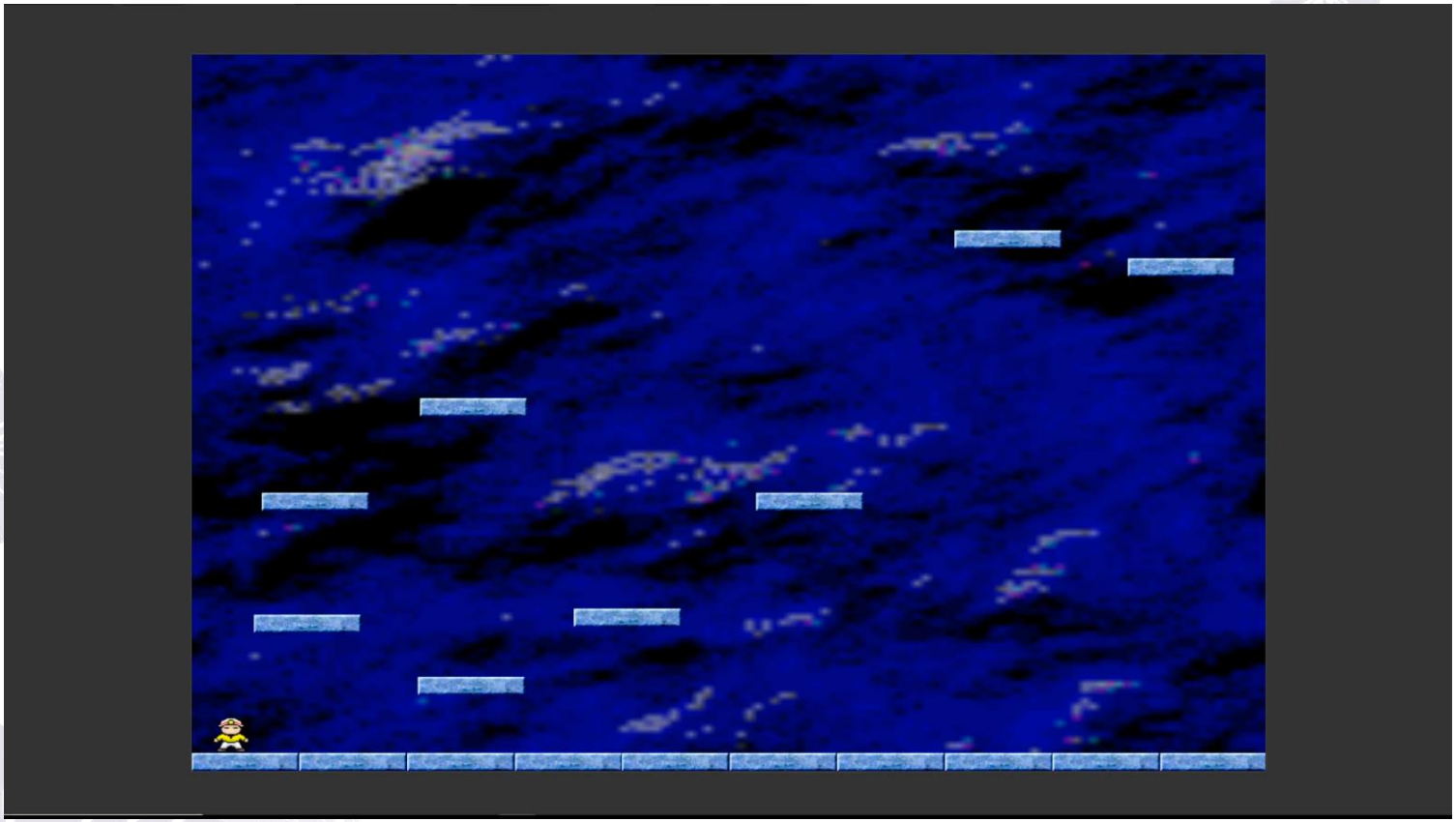
Create walk animation

- Use node.scaleX to deal with flipping
  - In PlayerController.ts

assets > Scripts > TS PlayerController.ts > PlayerController > update

```
44      update (dt){
45          this.node.x += this.playerSpeed * this.moveDir * dt;
46          this.node.scaleX = (this.moveDir >= 0) ? 1 : -1;
47          if(this.getComponent(cc.RigidBody).linearVelocity.y != 0)
48              this.fallDown = true;
49          else
50              this.fallDown = false;
51
52          this.playerAnimation();
53      }
```

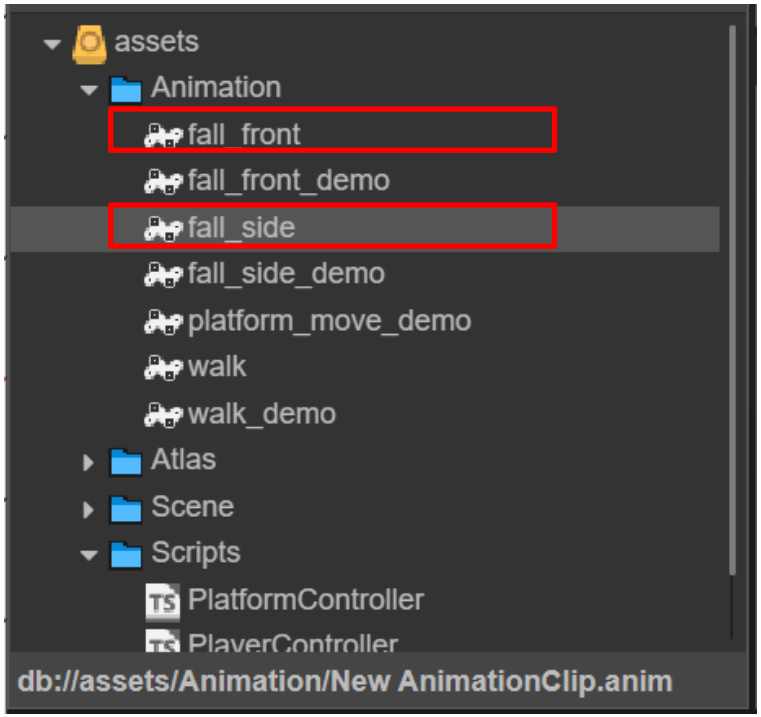
Create walk animation




Your
turn!
C++
PASCAL

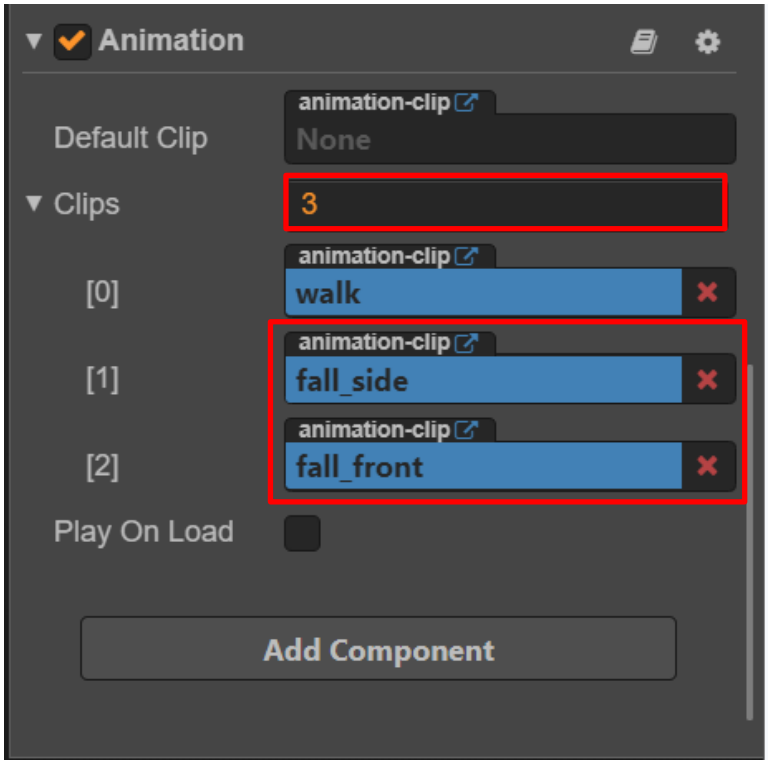
Create fall animation

- Step 1: Create two empty animation clips, named “fall_front” and “fall_side”



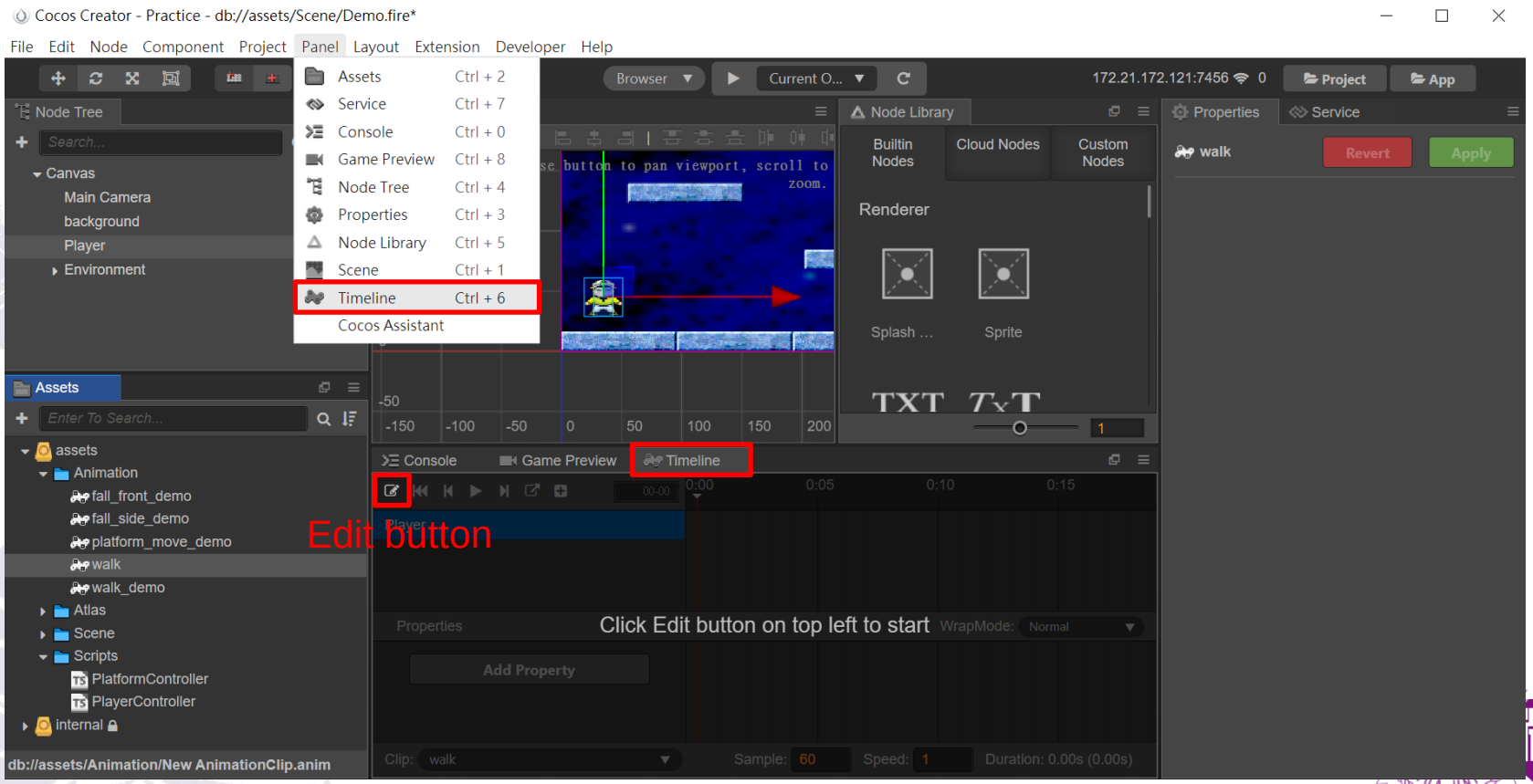
Create fall animation

- Step 2: Put two animation clips to the animation component of “Player” node
  - Change the # of clips to 3



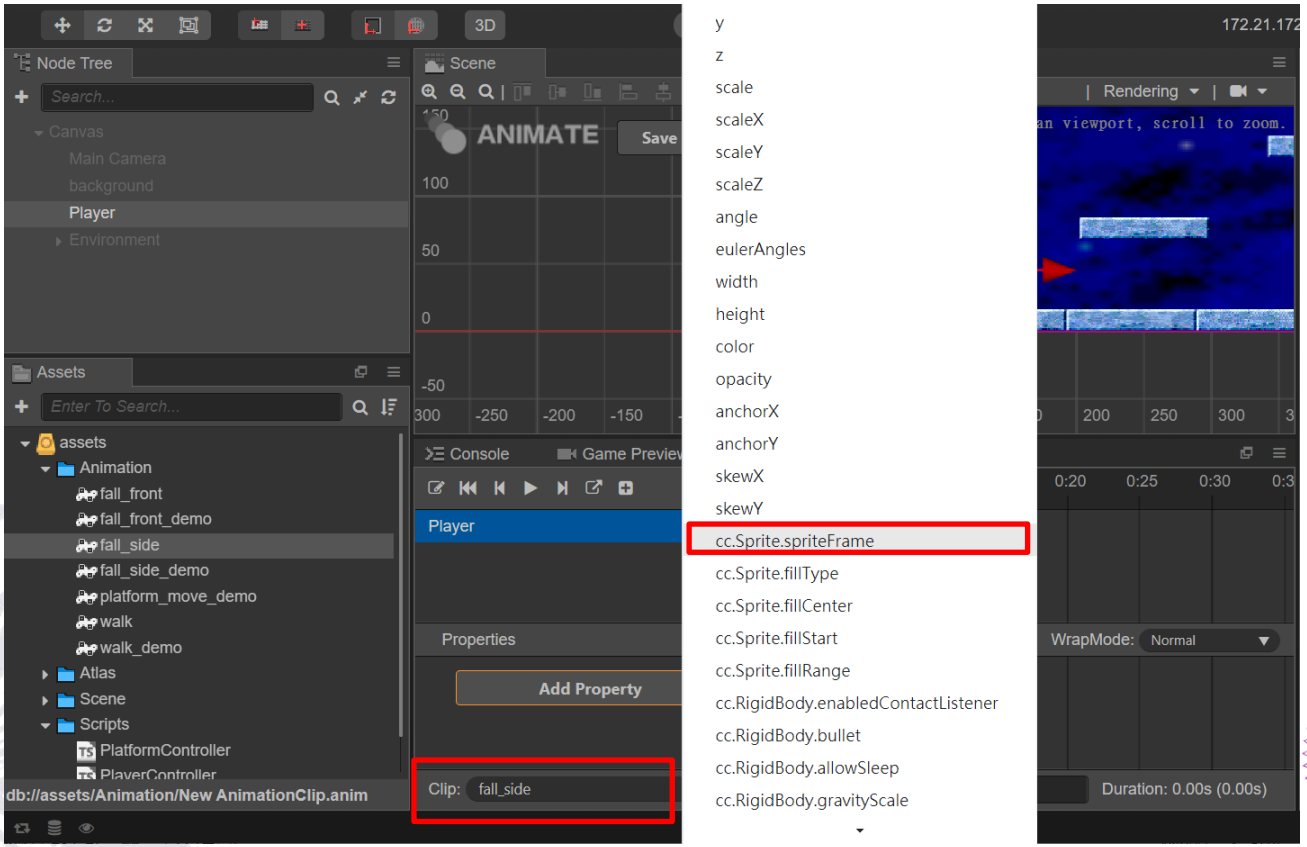
Create fall animation

- Step 3: Open animation editor (Timeline)



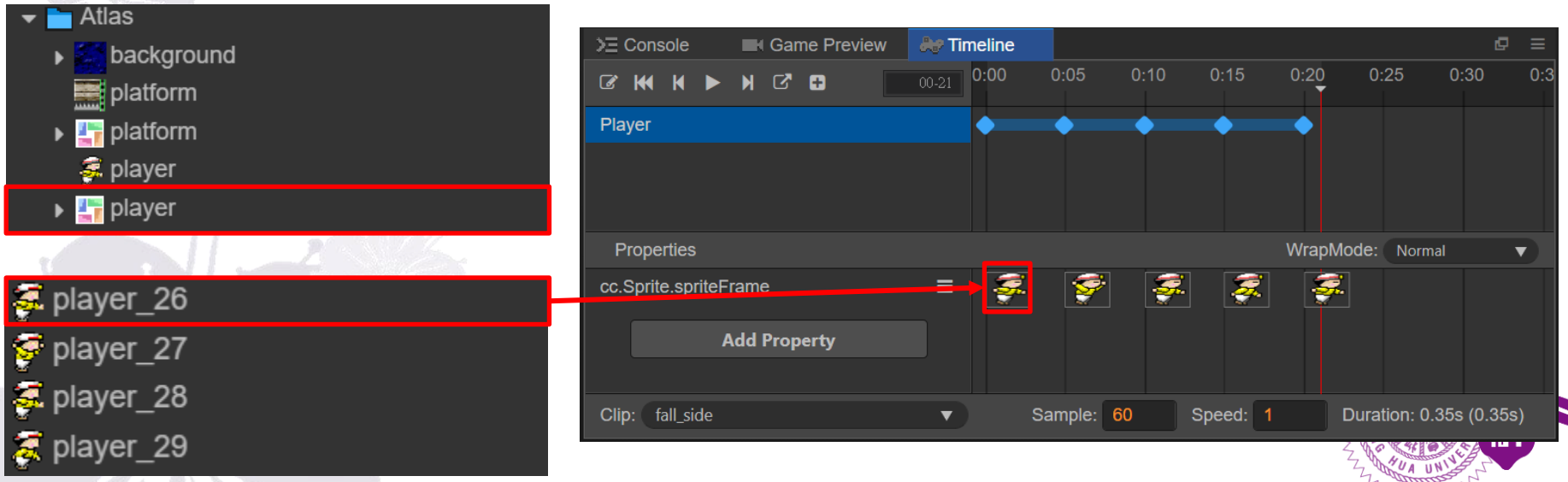
Create fall animation

- Step 4: Choose the “fall_side” animation clip
- Step 5: Add Property→cc.Sprite.spriteFrame



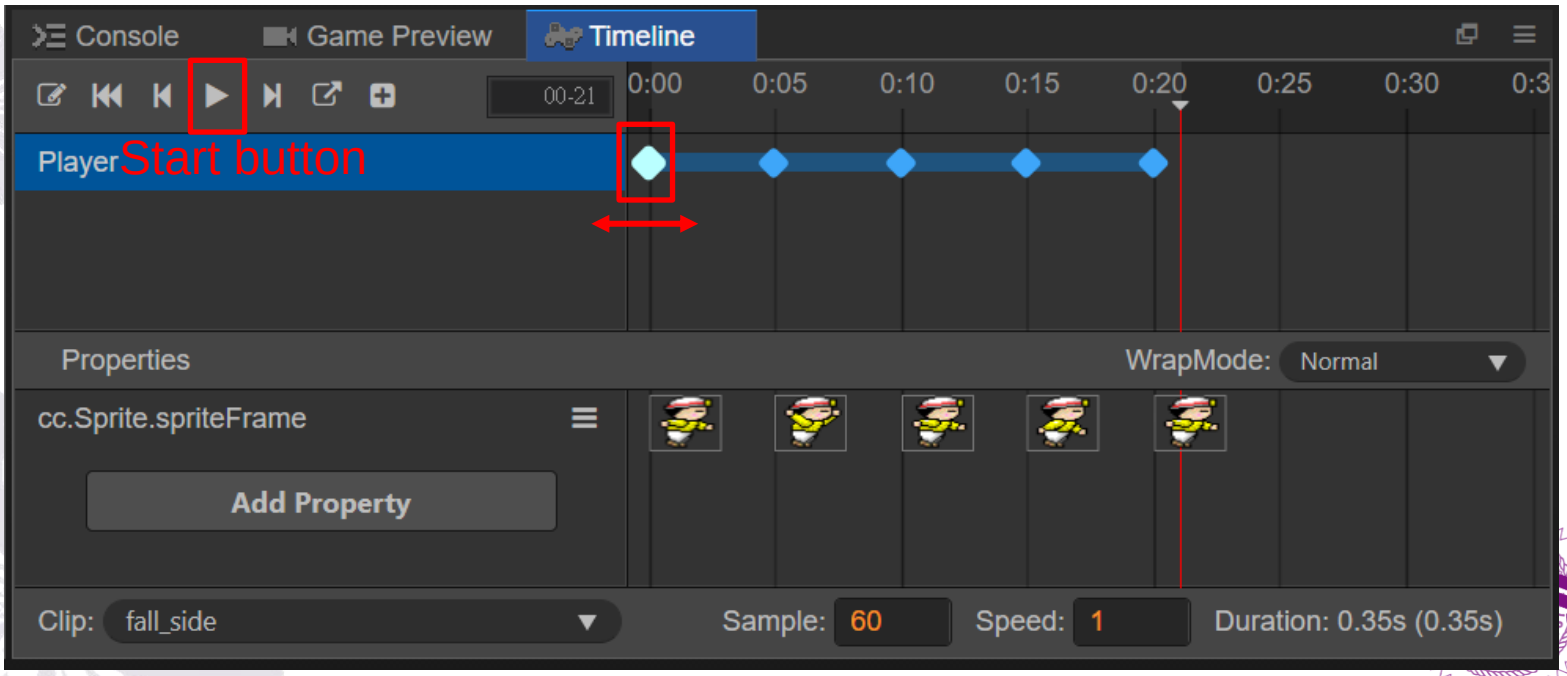
Create fall animation

- Step 6: Drag the sprite to animation editor
  - Sprite location: Atlas/player
  - Sprite order: player_26→ player_ 27→ player_ 28→ player_ 29→ player_ 26



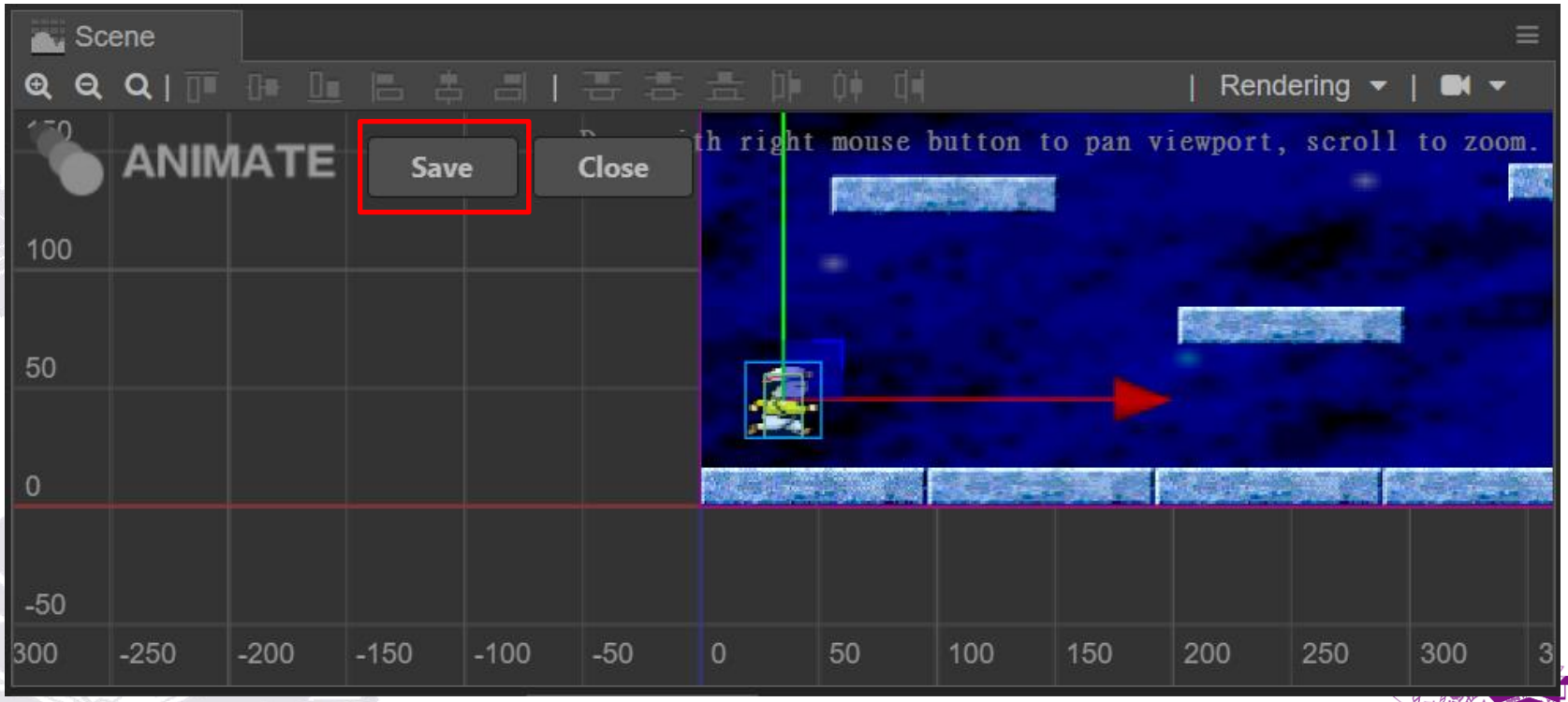
Create fall animation

- Step 7: Adjust the time interval between each sprite
  - Drag keyframe directly
  - Press the start button to preview animation



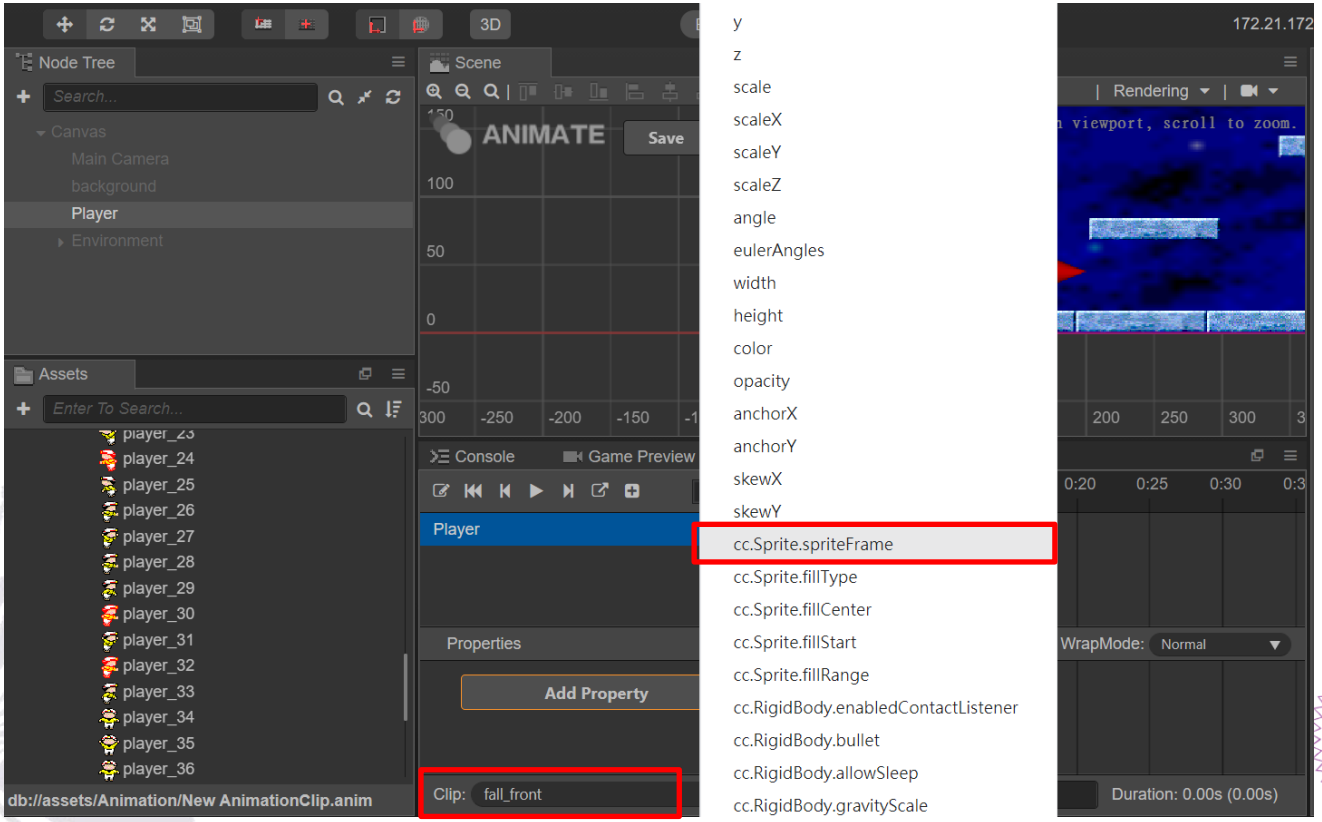
Create fall animation

- Step 8: Save the changes



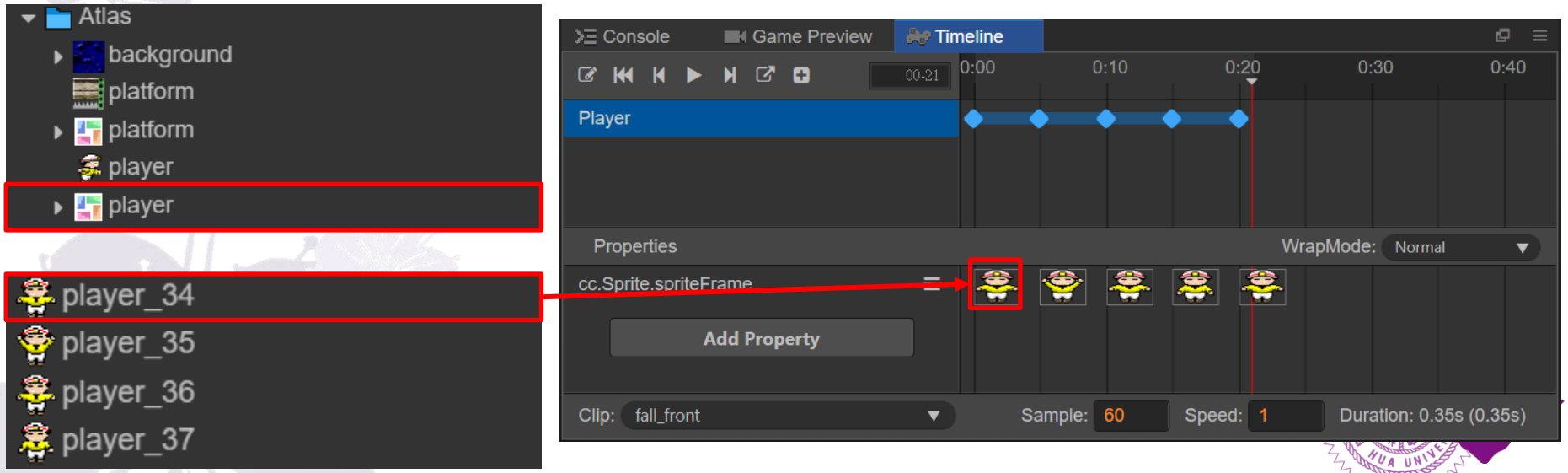
Create fall animation

- Step 9: Choose the “fall_front” animation clip
- Step 10: Add Property→cc.Sprite.spriteFrame



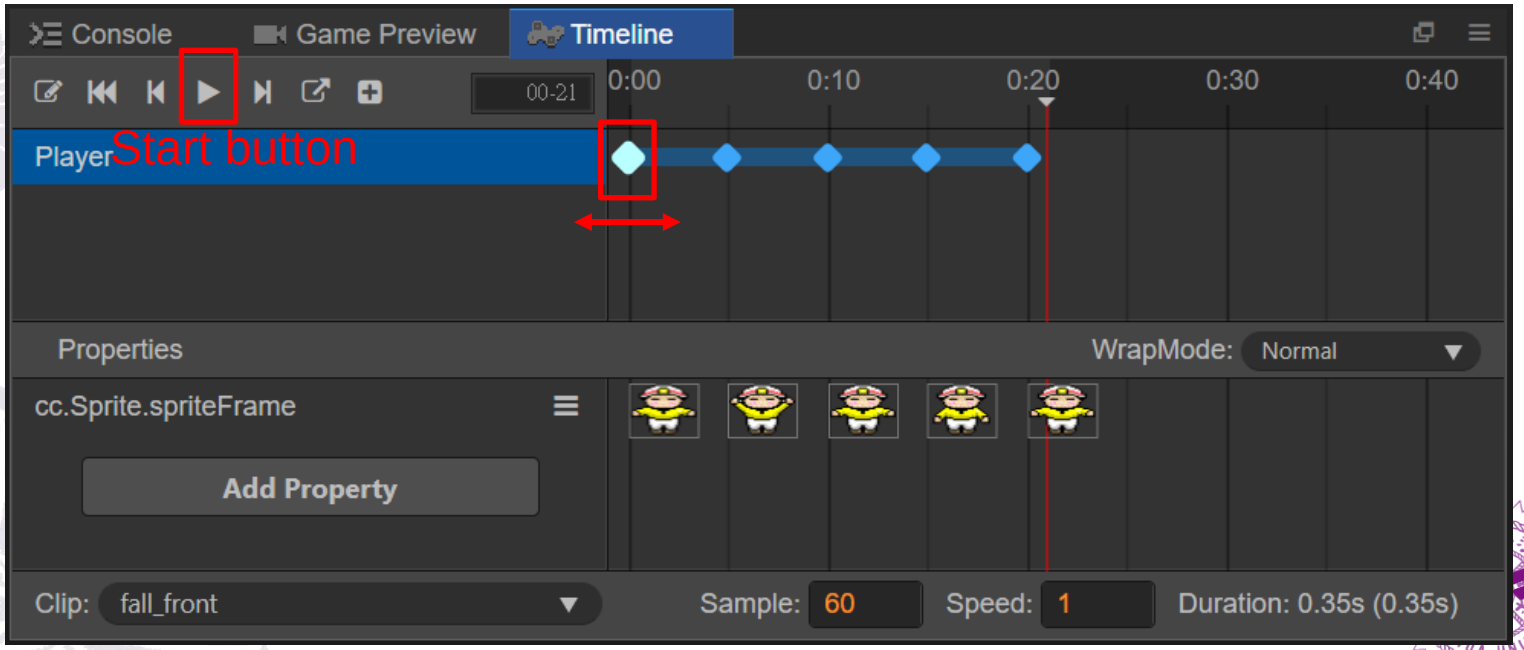
Create fall animation

- Step 11: Drag the sprite to animation editor
  - Sprite location: Atlas/player
  - Sprite order: player_34→ player_ 35→ player_ 36→ player_ 37→ player_ 34



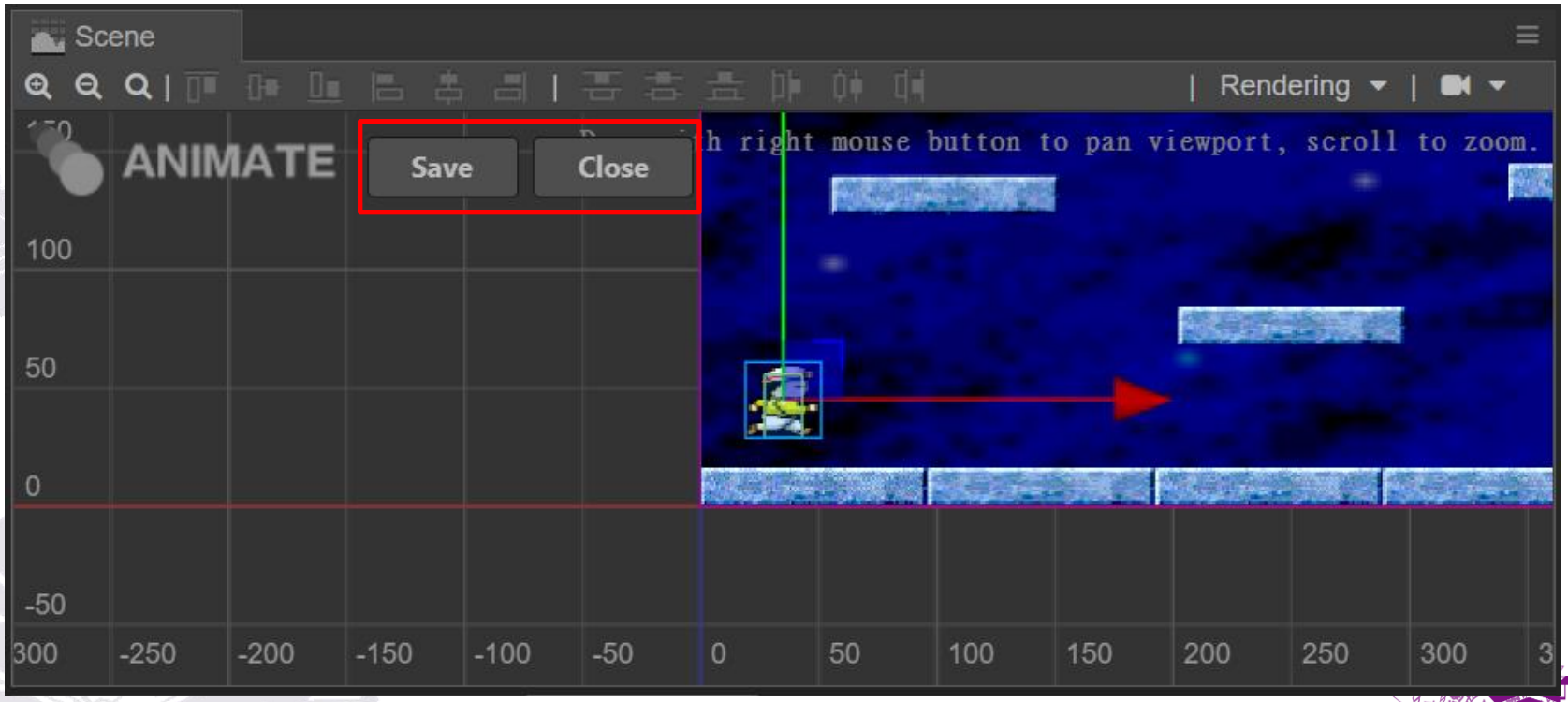
Create fall animation

- Step 12: Adjust the time interval between each sprite
  - Drag keyframe directly
  - Press the start button to preview animation



Create fall animation

- Step 13: Save the changes and close the editing status



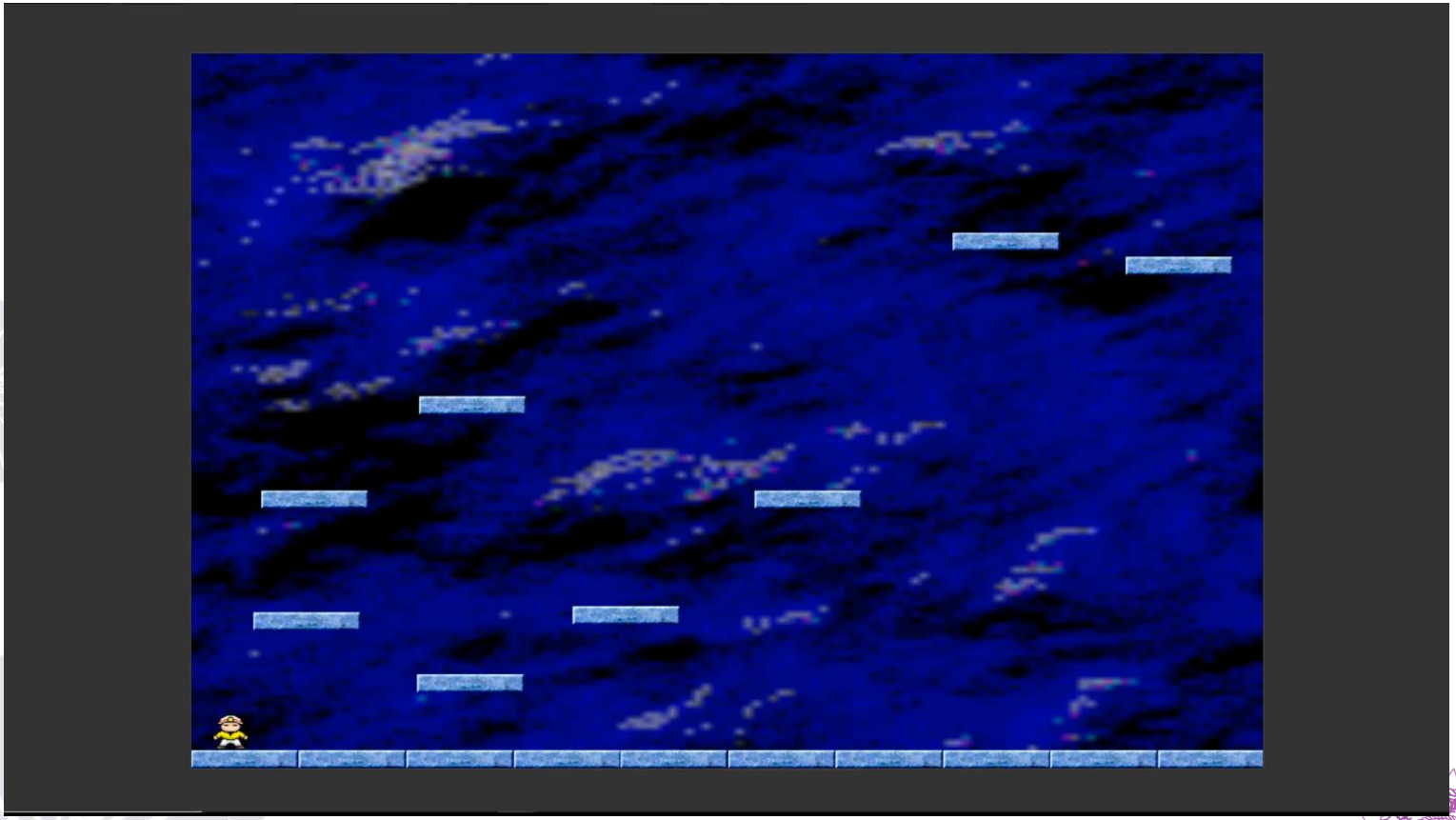
Create fall animation

- Step 14: Play animations with the script

assets > Scripts > TS PlayerController.ts > PlayerController > playerAnimation

```
100     public playerAnimation(){
101         if(this.fallDown == true){
102             // ========== TODO ==========
103             // 1. Play fall_front animation (Checked the animation is playing or not and moveDir=0)
104             // 2. Play fall_side animation (Checked the animation is playing or not and moveDir != 0)
105             if(this.moveDir == 0 && !this.anim.getAnimationState("fall_front").isPlaying){
106                 this.anim.play("fall_front")
107             }
108             else if(this.moveDir != 0 && !this.anim.getAnimationState("fall_side").isPlaying){
109                 this.anim.play("fall_side");
110             }
111         }
```

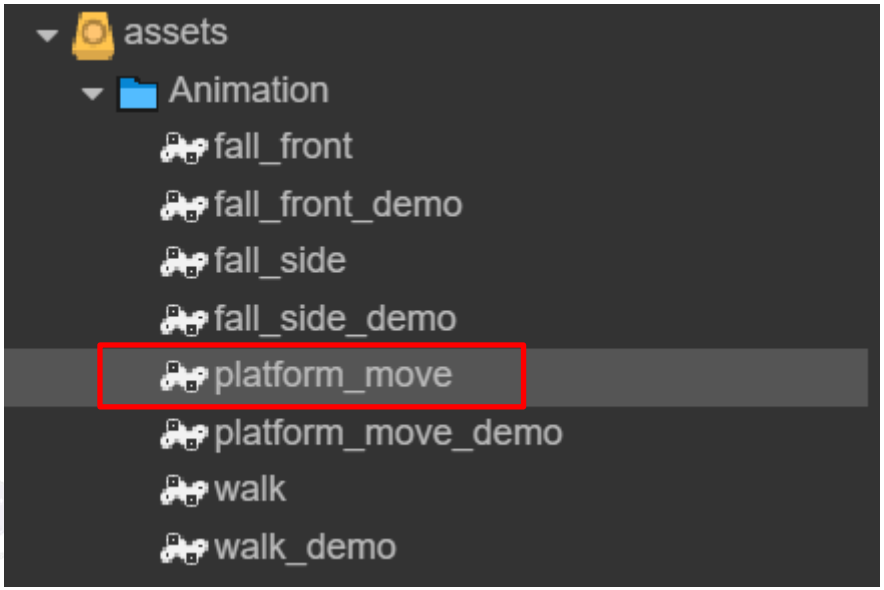
Create fall animation




Your
turn!
C++
PASCAL

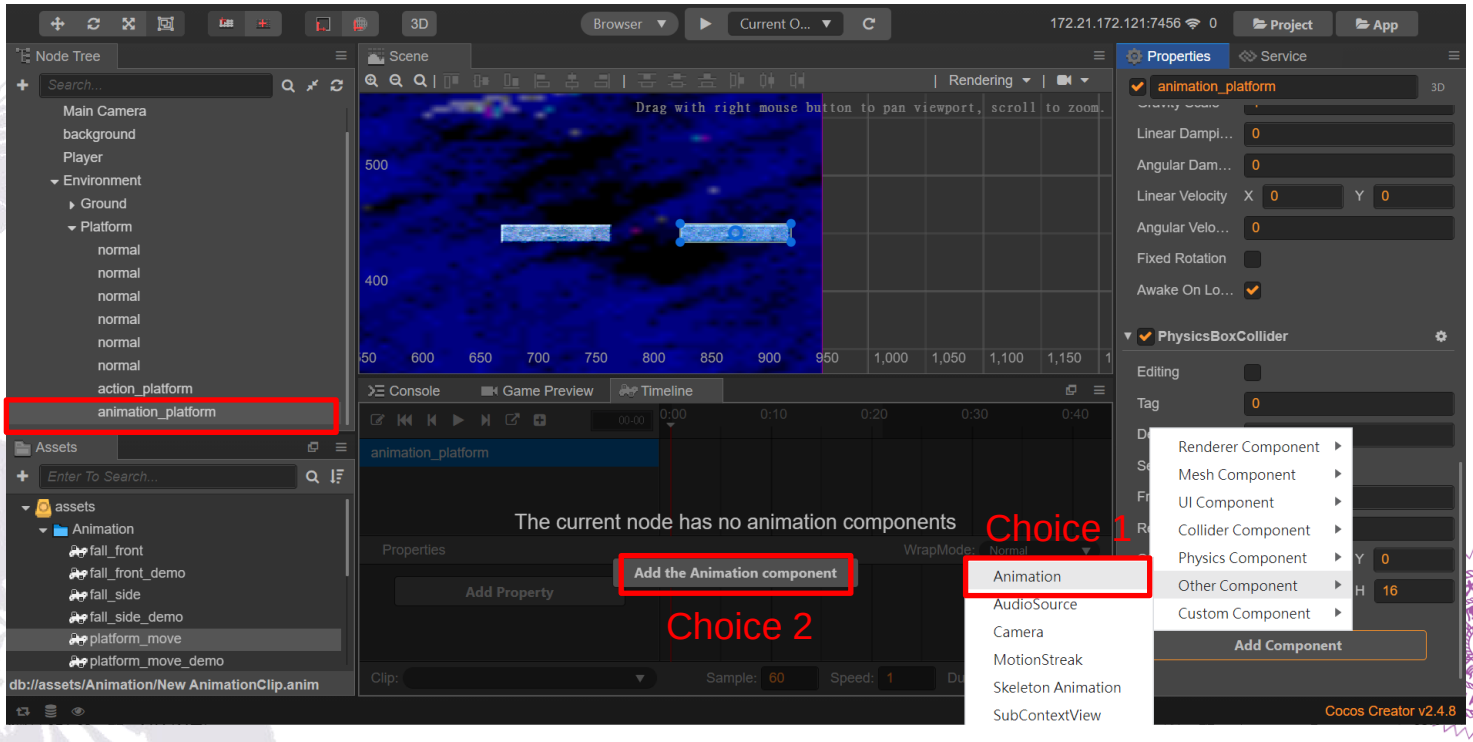
Create moving platforms animation

- Step 1: Create an animation clips, named “platform_move”



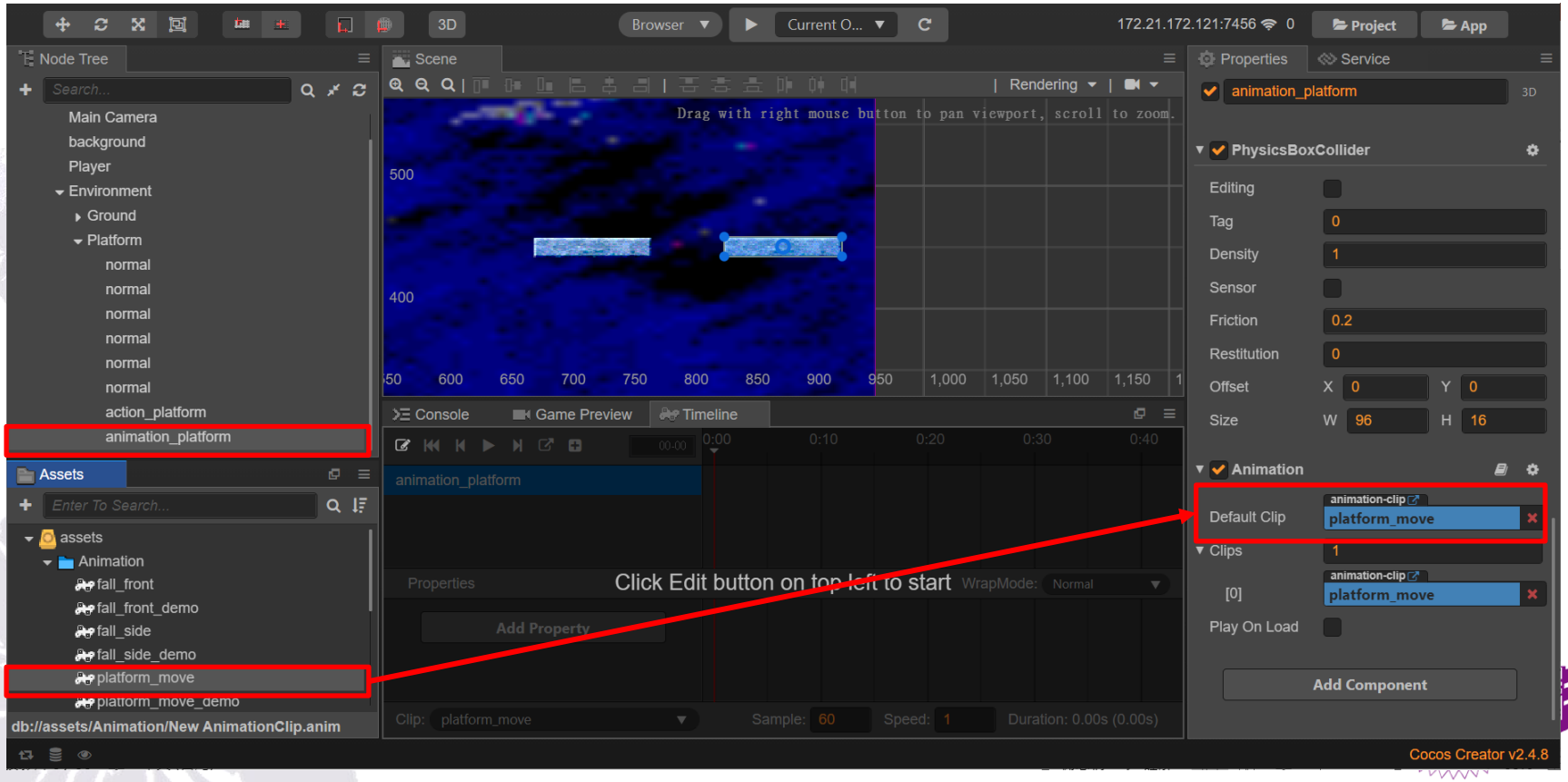
Create moving platforms animation

- Step 2: Create animation component
  - Choose the Environment/Platform/animation_platform node
  - Add component → Other component /Animation
  - Or click “Add the Animation component” at Timeline panel



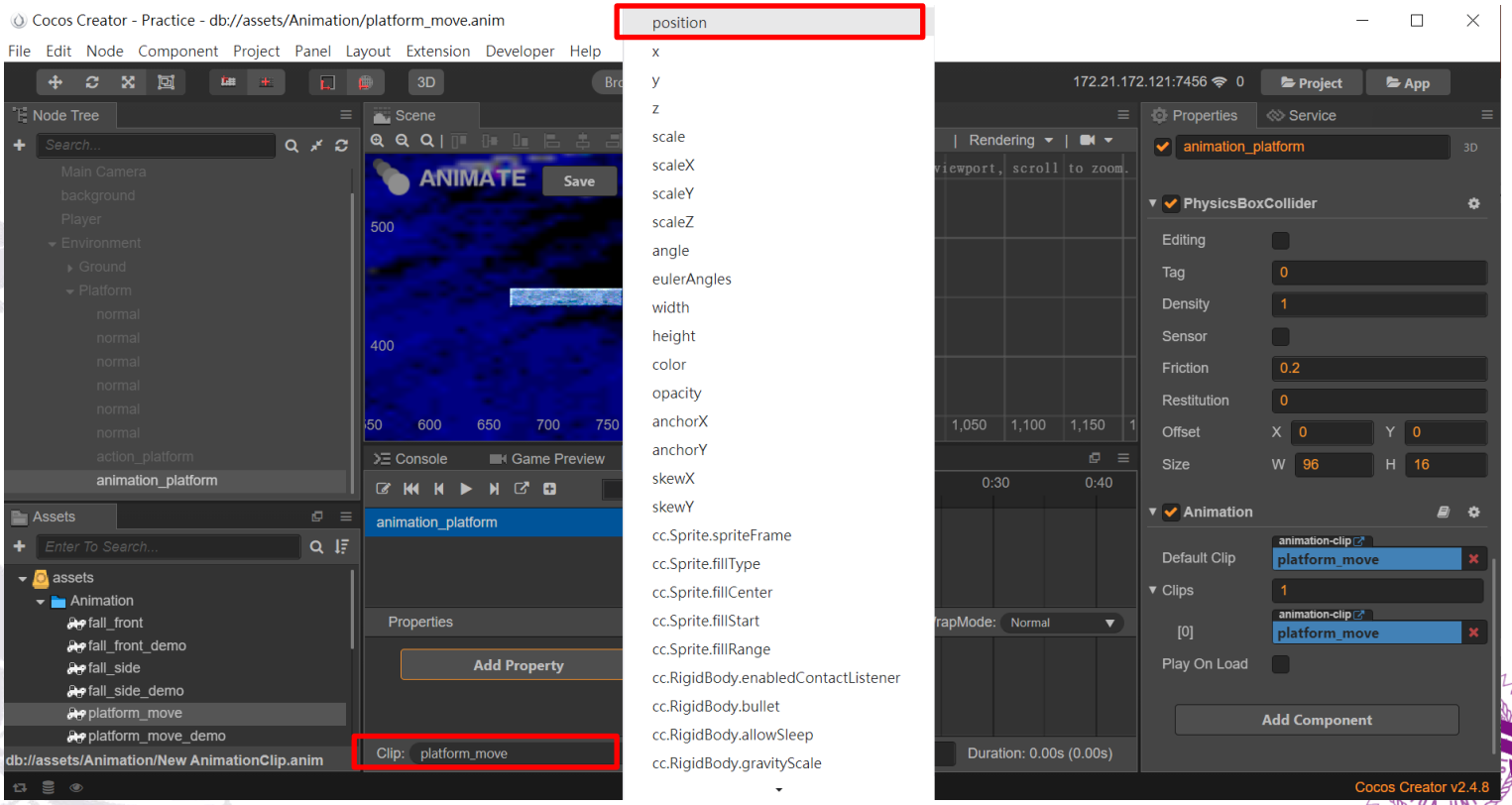
Create moving platforms animation

- Step 3: Put the “platform_move” animation clip to the animation component



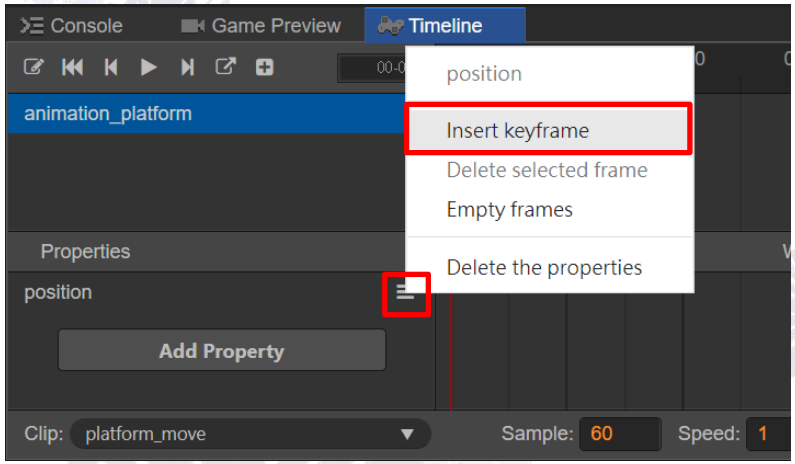
Create moving platforms animation

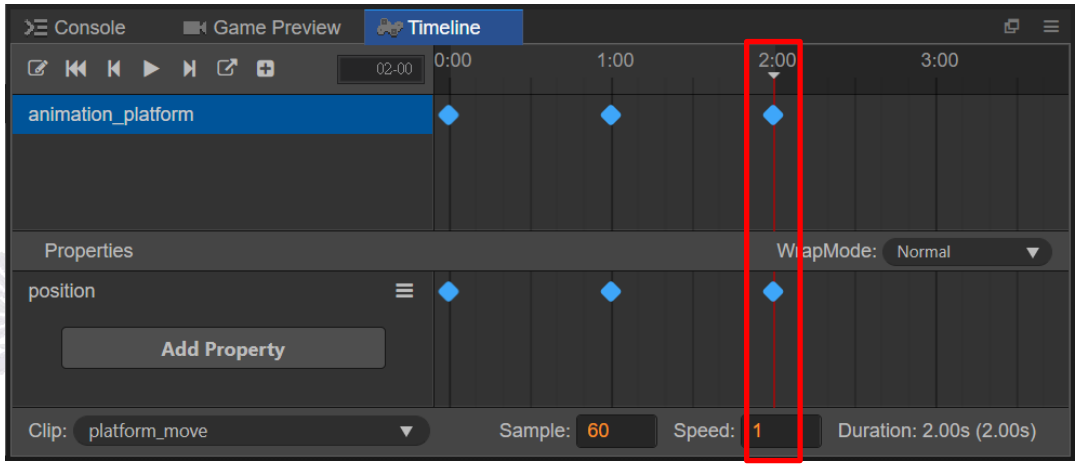
- Step 4: Choose the “platform_move” animation clip
- Step 5: Add Property→position



Create moving platforms animation

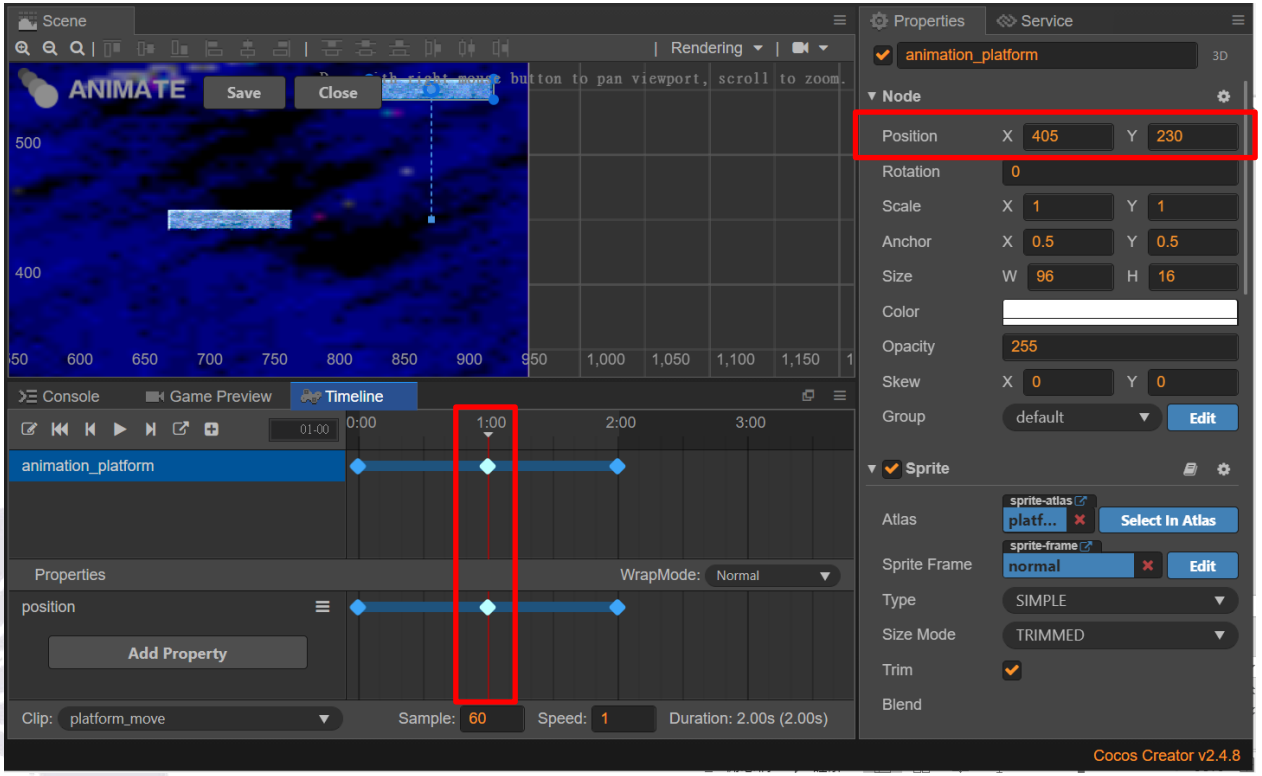
- Step 6: Insert 3 keyframes
  - Drag the red line to the expected time and insert the keyframe





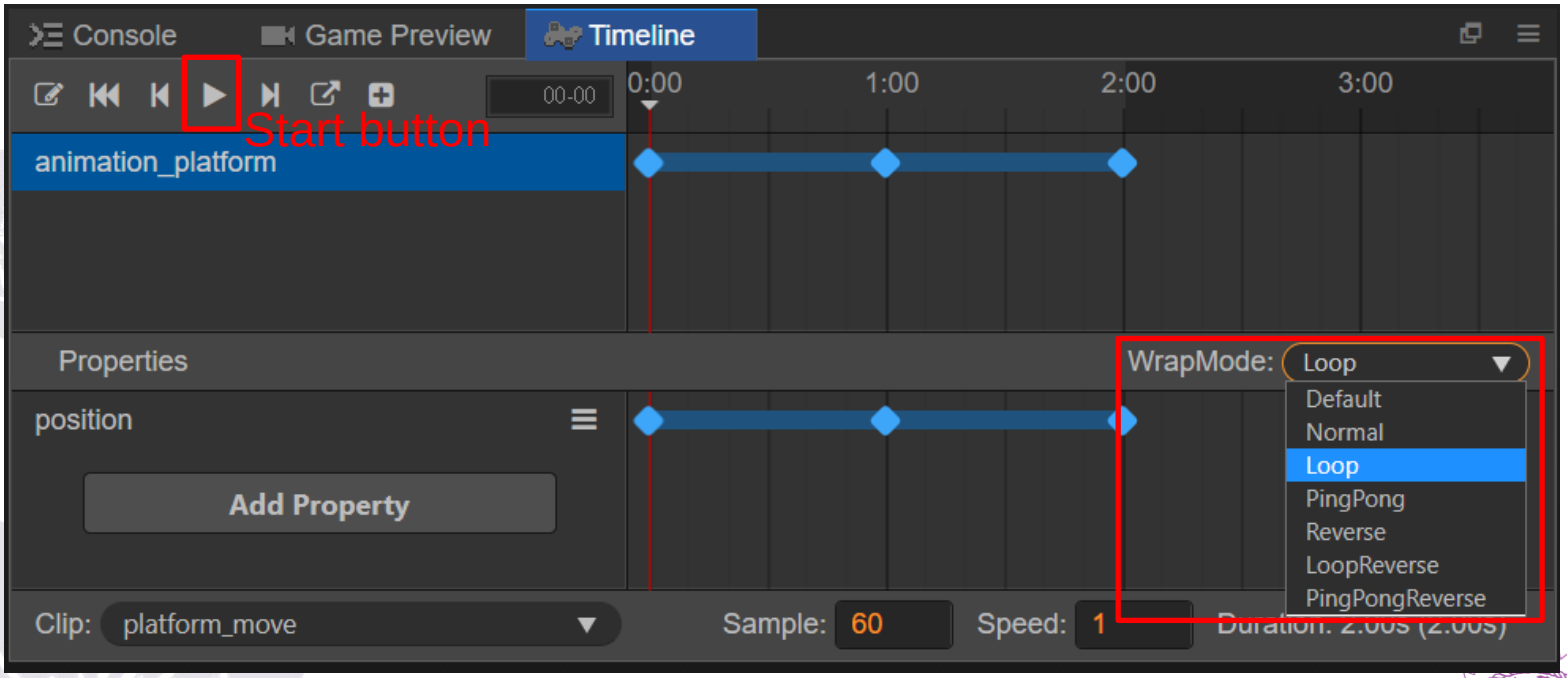
Create moving platforms animation

- Step 7: Adjust the platform position at each key frame
  - position: (405, 130) →(405, 230) → (405, 130)



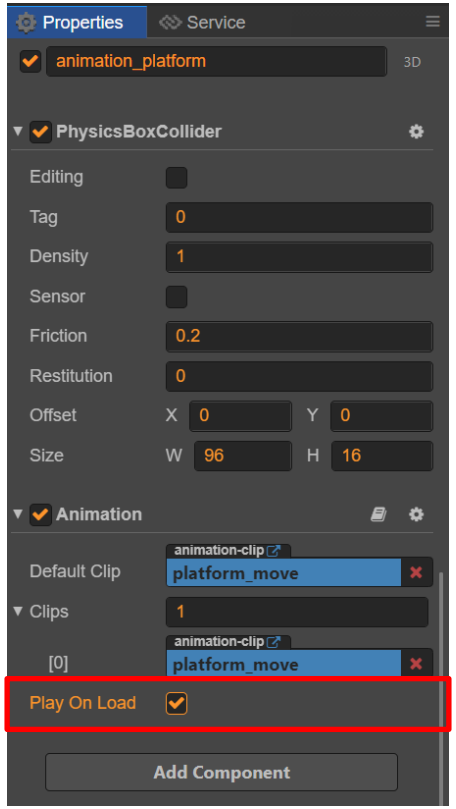
Create moving platforms animation

- Step 8: Change WarpMode to Loop
  - Press the start button to preview animation



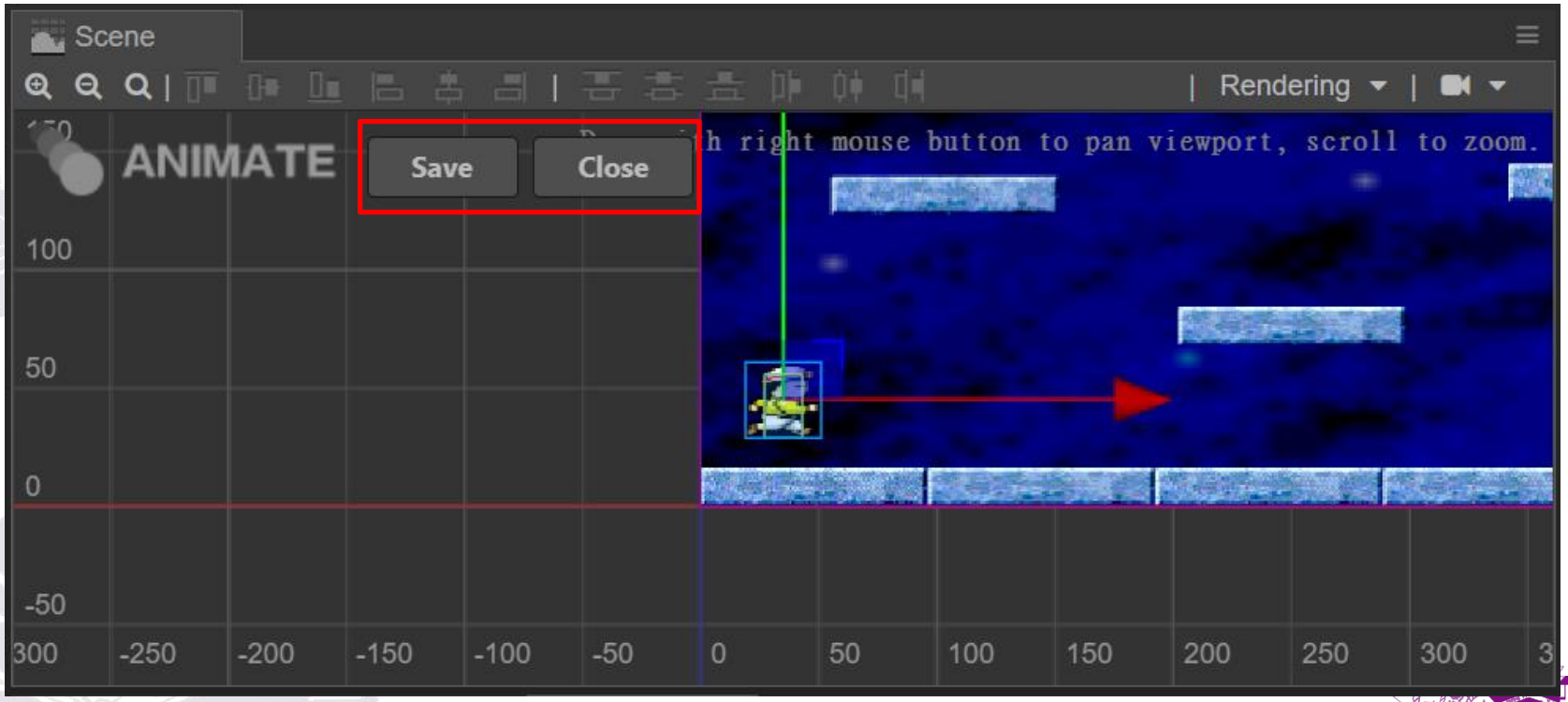
Create moving platforms animation

- Step 9: Click “Play on Load” at the animation component of “animation_platform”

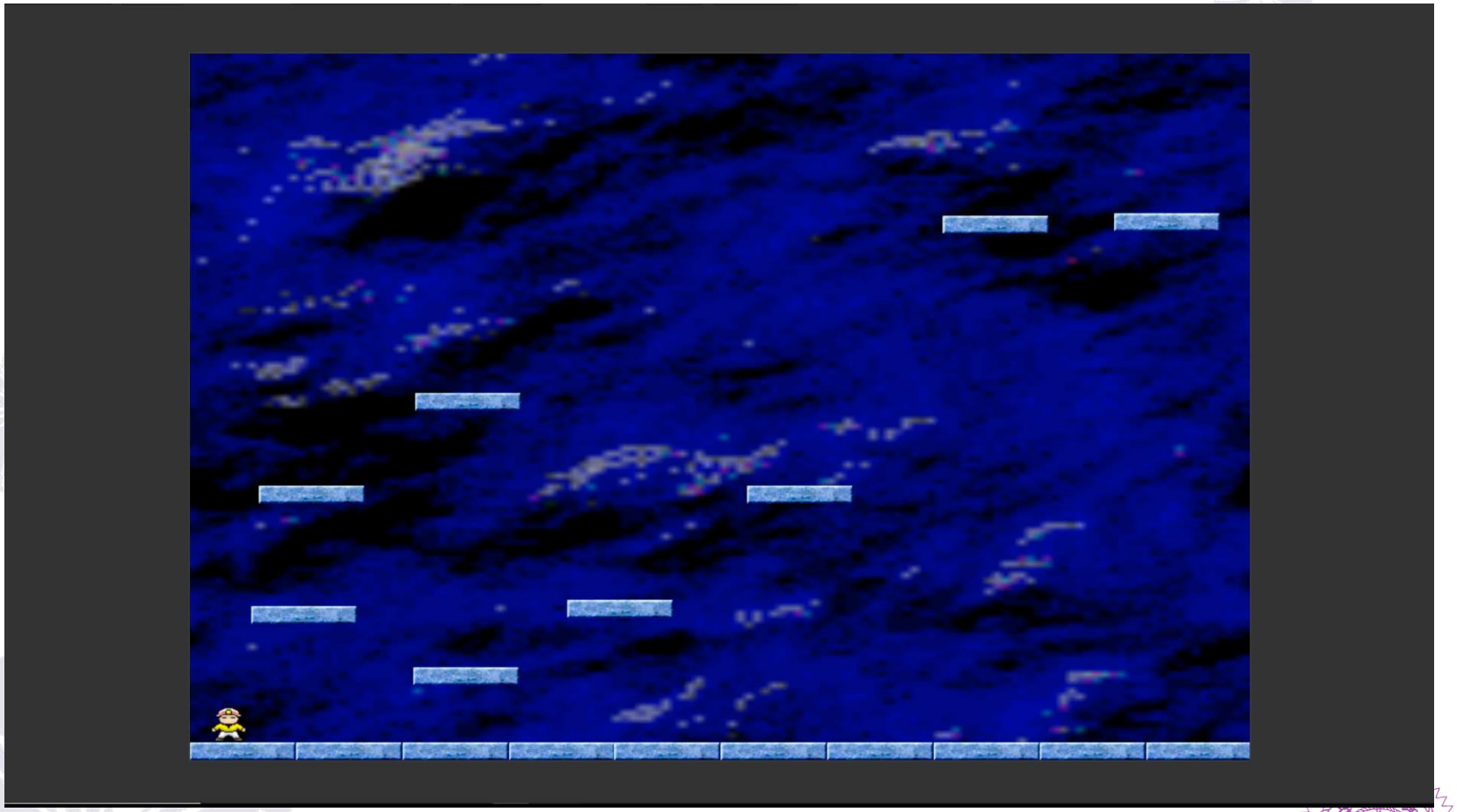


Create moving platforms animation

- Step 10: Save the changes and close the editing status



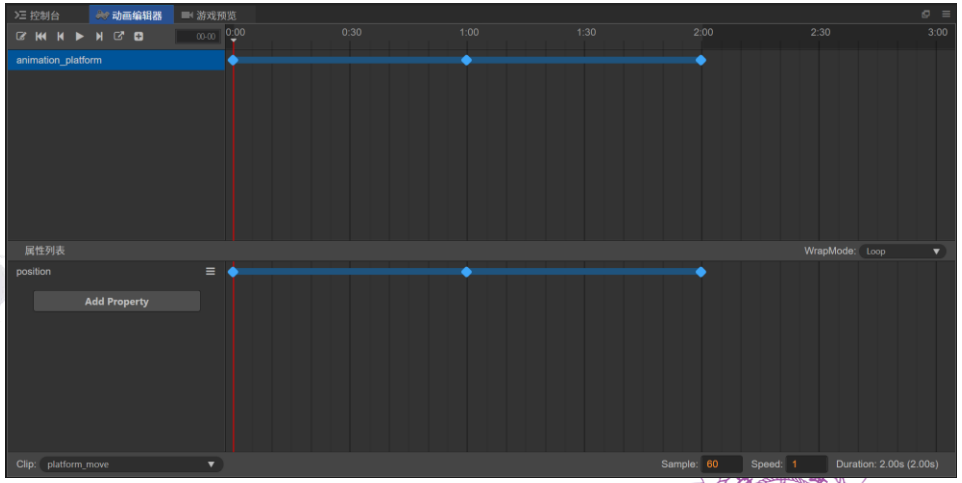
Create moving platforms animation



Create moving platforms animation

- The result of the action system and animation is the same
- Using action system to do the activity which is affect the game system
- Using animation to do the activity which is a show or appearance of the game

```
public platformMove(){
    let action: cc.Action;
    var seq = cc.repeatForever(
        cc.sequence(
            cc.moveTo(1, 250, 230),
            cc.moveTo(1, 250, 130)
        ));
    this.node.runAction(seq);
}
```




Your
turn!
C++
PASCAL