05 Keyboard Input

Description

In this lesson students will see how to use the keyboard as an input for p5.play. It will start with an opening exploration where students discover when the different keyboard events are triggered.

Students will also learn how to move a sprite using the keyboard and have a chance to practice with a partner.

Before class, print a copy of the note catcher for each table group of students.

Objectives

- I can use the keyboard to move a sprite.
- I can keep a sprite in the screen using static colliders.

Brain-Starter + Hook (5 min)

Prompt:
“In the last lesson you learned how to control sprites using the mouse.
How else would you want to interact with a p5 game?”

Have students turn and talk.

Have groups share out. Keyboard will likely be the first answer.

Push their thinking some, possible followups:
Q: “What keyboard keys to most games use?
A: WASD, arrow keys,
Q: “What about for two player games?
A: IJKL (common, but students might struggle with this one)
Q: “Besides the keyboard and mouse, are there any other ways we can control games?”
A: Game controllers!

P5.play has three different ways to get player input.

- mouse
- keyboard
- game controllers

Today we are going to zoom in on keyboard input.

Exploration (10 min)

Direct students to the exploration website
source code | permalink

In groups, have students interact with this page, then try and put into common language when each event is triggered.

Sentence starter:
[Event Type] is triggered when... but not when...

Ex. “Press is triggered when the key is pressed, but not when it is held down.”

Do one or two as a group if students are struggling.
As a group, have students use the note catcher below to record their findings. Record on a poster or slide to create an anchor chart for students to reference later.

Live Code #1 (10 min)

Start Code
Final Code

```
//PROVIDE: starter code with a preload, setup and draw function;
let p1, kirby;

function preload() {
  kirby = loadAnimation(
    "assets/kirby_walking_0001.png",
    "assets/kirby_walking_0002.png",
    "assets/kirby_walking_0003.png",
    "assets/kirby_walking_0004.png",
    "assets/kirby_walking_0005.png",
    "assets/kirby_walking_0000.png"
  );

  //if you are using replit, you can load an animation like this:
  //starAni = loadAnimation('./assets/pixil-frame-1.png', 8);
}

function setup() {
  createCanvas(800, 400);
  //REVIEW: How can I create a sprite? How can I make the appearance be the kirby
animation?
  p1 = new Sprite(kirby);

  //PLAY: See the sprite in the middle of the screen, with animation applied
  //Too Fast, review how to to slow down an animation.
  p1.animation.frameDelay = 8;

  console.log(kb);
  //debugger;
}

function draw() {
  //REVIEW: What does this do? What happens if I delete it?
  clear();
```

```
  //VERSION 1 : Make the sprite move left
  // if(kb.pressing('right')){
  //   p1.vel.x = 1; //ASK: How could I make Kirby move faster?
  // }

  //OOPS! How to I get Kirby to stop?
  //VERSION 2 : right only when pressing
  // if(kb.pressing('right')){
  //    p1.vel.x = 1; //ASK: How could I make Kirby move faster?
  // } else {
  //   p1.vel.x = 0;
  // }

  //VERSION 3 : left and right
  //ASK : I want kirby to als be able to move left and right, anyone got an idea?
  // if (kb.pressing("right")) {
  //   p1.vel.x = 5; //ASK: How could I make Kirby move faster?
  // } else if (kb.pressing("left")) {
  //   //ASK : what should the velocity be to make Kirby move left?
  //   p1.vel.x = -5;
  // } else {
  //   p1.vel.x = 0;
  // }

  //ASK : What changes would you want to make? Start/Stop Animation, flip directions.
  //VERSION 4 : Bells and whistles!
  if (kb.pressing("right")) {
    p1.vel.x = 5; //ASK: How could I make Kirby move faster?
    p1.animation.play();
    p1.mirror.x = false;
  } else if (kb.pressing("left")) {
    //ASK : what should the velocity be to make Kirby move left?
    p1.animation.play();
    p1.vel.x = -5;
    p1.mirror.x = true; //flip direction, OOPS! : now he won't turn around
  } else {
    p1.vel.x = 0;
    p1.animation.stop(); //stop animation when not moving
    p1.animation.frame = 0; //set to the first frame
  }

  p1.debug = mouse.holding();
}
```

Task #1 (10 min)

Starter Code for students
Final Code for reference

Directions

- 1 - Create a new sprite for p1, give it the flappy animation.
- 2 - Make the bird move up and down with the keyboard.
- Challenge - Stop and Stop animation when the keys are being pressed.
- Boss Level!
    - In `setup()` - Create a second sprite for a second player.
    - give player 2 the same animation, but point it the other way.
    - in `draw()` - Make this new sprite also move up and down but different keys.
    - HINTS:
        - It is common to use JIKL for player 2 movement.
        - The a key is set to "left" in p5.play, and is usually used for player 1
        - The j key is set to "left2" in p5.play, and can be used to move player 2

Live Code #2 (10 min)

Return to the where you left off in Live Code #1. Add this to the setup to create a wall for Kirby.
Final

```
function setup() {
  createCanvas(800, 400);

  //SAY : on day 1, you create sprites that didn't move
  //ASK : how could we create an obstacle that stops kirby?
  //generate this will students!
  //VERSION 1 : something in the way
  obstacle = new Sprite(100, 200, 20, 50);
  obstacle.collider = 'static';

  //VERSION 2 : move the change the size of the obstacle to make it a wall at the left
side of the screen.
  //Generate the size and position of the wall with the students.
  //obstacle = new Sprite(-5, height/2, 10, height);
  //obstacle.collider = 'static';

  p1 = new Sprite(kirby);
  p1.animation.frameDelay = 8;
}
```

Task #2 (10 min)

Student return to their code for Task #1.

Directions:

Add a “ground” sprite that keeps the bird from exiting the screen on the bottom.
Birds can fly as high as they like!

Students should submit code at the end of the lesson so you can identify common mistakes and see where they are struggling.

Closing (5 min)

Turn and Talk: We have now seen how to move a sprite with a keyboard and the mouse.
Which type of input do you like most? Which type do you think you will use in your first game?

Keyboard Events!

Presses	Pressing	Released
Presses is triggered when... but not when... Code	Pressing is triggered when... but not when... Code	Released is triggered when... but not when... Code
Holds	**Holding**	**Held**
Holds is triggered when... but not when... Code	Holding is triggered when... but not when... Code	Held is triggered when... but not when... Code