CSC 496 Fall 2023
Lab 1: Pokémon Fight Game in SwiftUI

Dr. Si Chen

September 20, 2023

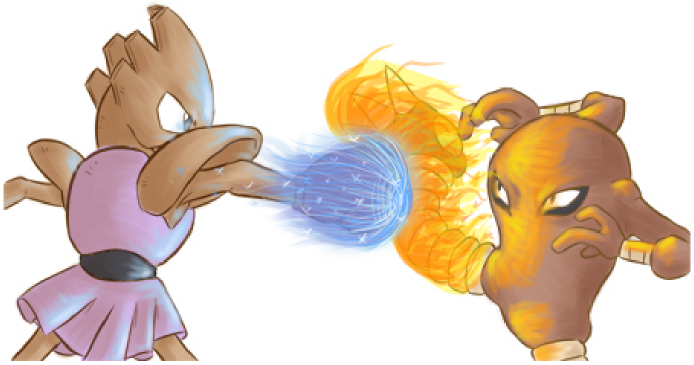

Figure 1: A Pokémon Fight Game

Introduction

The goals of this lab are:

- Understand the basic use of SwiftUI for UI development.
- Implement a simple game logic using state management in SwiftUI.
- Learn to use basic animations in SwiftUI.

Our course webpage: https://www.cs.wcupa.edu/schen/ios23/

Lab Instructions

Objective

Develop a simple Pokémon fight game where two randomly selected Pokémon will fight each other. The game ends when one of the Pokémon's HP falls below zero.

Key Concepts to Cover

- SwiftUI for UI development
- State management using `@State`
- Basic game logic

Requirements

1. Display two random Pokémon images side by side.
2. Each Pokémon starts with 1000 HP.
3. Pokémon take turns to attack each other.
4. Pressing the ”Attack” button deals a random amount of damage (1-100) to the opponent.
5. **Note:** Only the attacking Pokémon should shake when the ”Attack” button is pressed.
6. Display the initial HP (Hit Points) of each Pokémon as 1000, and dynamically update it after each attack turn. The HP should decrease based on the random damage dealt during an attack.
7. The game ends when one Pokémon's HP falls below zero.
8. Display the winner of the battle.
9. Pressing the ”Replay” button should reset the game.

Evaluation Criteria

- Correctly implemented game logic (60%)
- Proper use of SwiftUI for layout and UI components (10%)
- Correct state management (20%)
- Code readability and comments (10%)

Hints..

- **Displaying Battle Status:** To display the battle status, the code uses the `@State var result: String = "Let's fight!"`. Look into modifying this variable to display different battle statuses.

```
@State var result: String = "Let's fight!"
Text(result).padding()
```

- **Randomly Selecting Pokémon:** The code uses two random integers to select which Pokémon images to display. You can use a similar technique to randomize other aspects of the game.

```
@State var randomNumber1 = Int.random(in: 1...100)
@State var randomNumber2 = Int.random(in: 1...100)
```

- **Shaking the Pokémon:** If you look at the `Button("Attack!")`, you'll see the code toggles the `pokemon1Shake` and `pokemon2Shake` variables to initiate shaking. You may want to change this logic so that only the attacking Pokémon shakes.

```
@State var pokemon1Shake: Bool = false
@State var pokemon2Shake: Bool = false
withAnimation {
    self.pokemon1Shake.toggle()
}
```

- **New Feature - Pokémon HP:** To implement a health feature for each Pokémon, you might want to add new `@State` variables to keep track of the HP for both Pokémon.

```
@State var pokemon1HP: Int = 100
@State var pokemon2HP: Int = 100
```

- **Calculating Attack Damage:** To calculate attack damage, you can use a similar function to `Int.random(in: 1...100)` to generate a random damage value.

```
let attackDamage = Int.random(in: 10...30)
```

Deliverables

Submit your whole project via compressed zip file to D2L under Lab 1. Your folder should include both code and project setting files (PokemonFight.xcodeproj).

Submission

- Check the lab due date on the course website. Late submissions will not be accepted.
- Submit your assignment to D2L directly.
- **No copy or cheating is tolerated**. If your work is based on others' or AI, please give clear attribution. Otherwise, you **WILL FAIL** this course.