

PLANTS vs. ZOMBIES

PRESENTATION PROJECT

LAB OOP GR1 – L2

--

CHICKEN TEAM

CREATED BY: CHICKEN TEAM

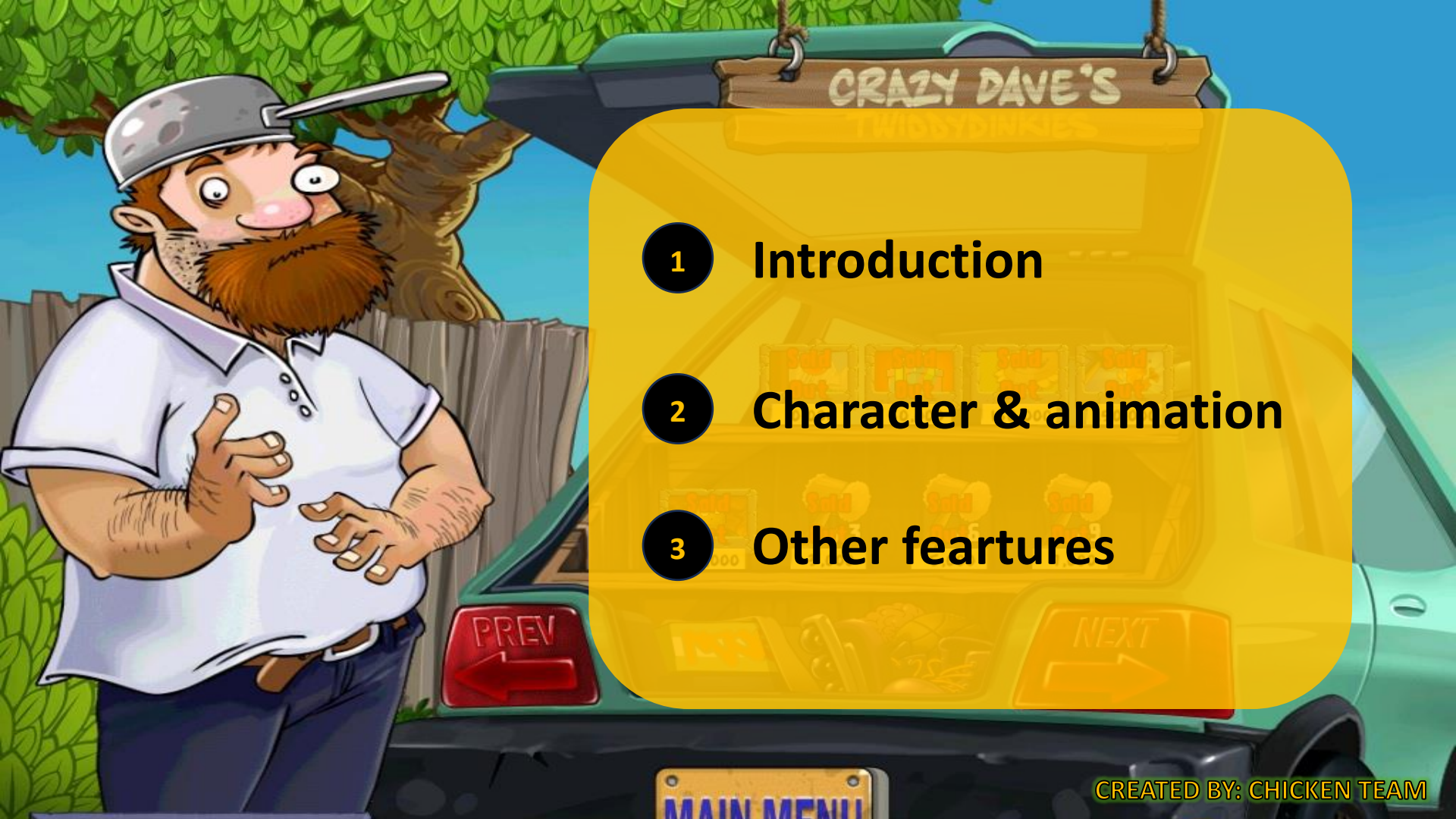
PLANTS vs. ZOMBIES

MEMBER

Nguyễn Hoàng Gia Ân
ITCSIU21161

Nguyễn Trọng Nguyên
ITITIU21236

Võ Minh Khang
ITCSIU21136



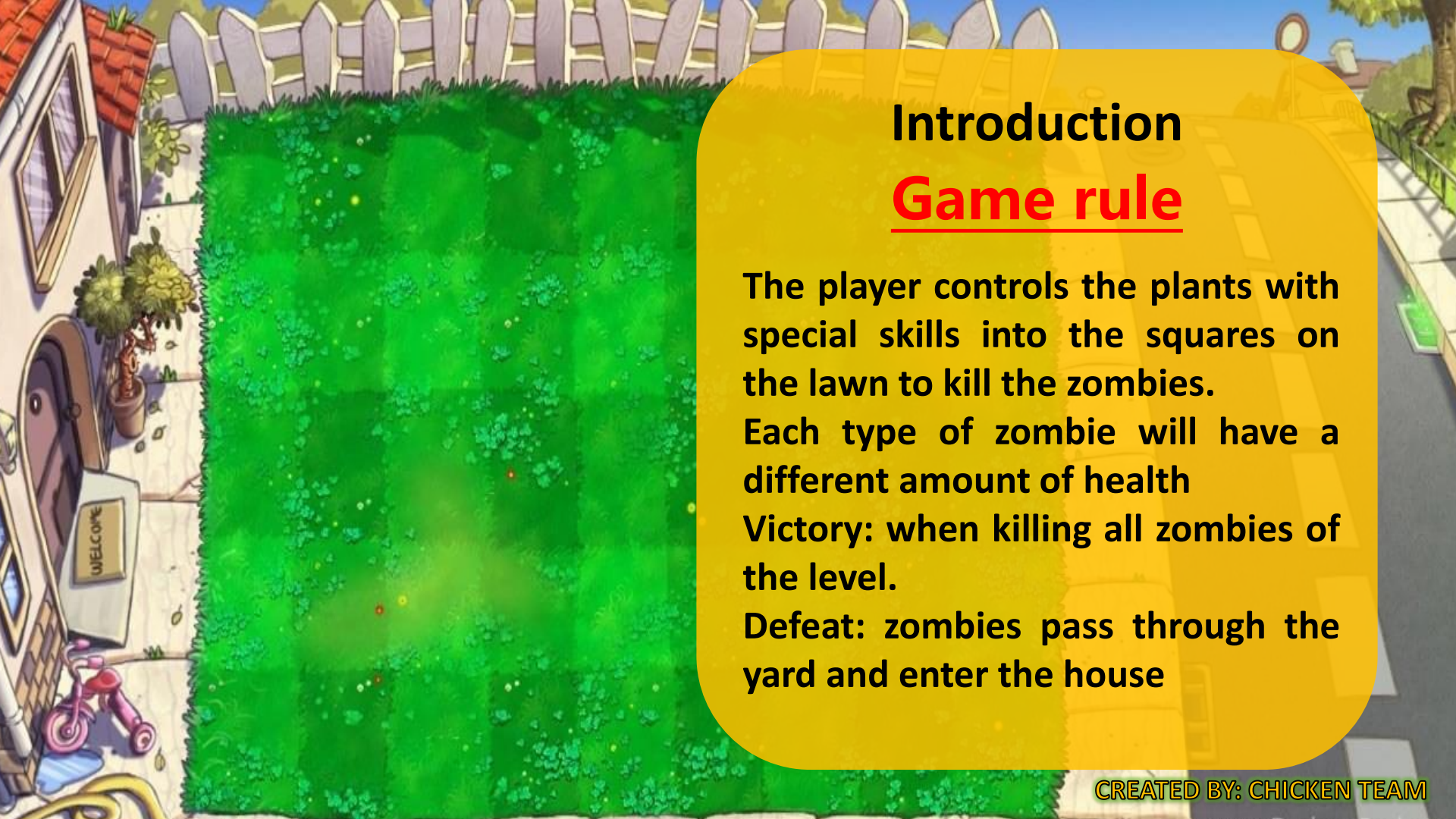
- 1 Introduction
- 2 Character & animation
- 3 Other feartures

The background of the slide is a colorful illustration from the game Plants vs. Zombies. It shows a suburban house on the left with a red roof, a window with a flower box, and a front door with a 'WELCOME' mat. A pink bicycle is parked on the sidewalk. A large, vibrant green lawn occupies the center, with a white picket fence in the background. The scene is set in a sunny, suburban neighborhood.

Introduction

PLANTS vs. ZOMBIES

The farmer faces a zombie epidemic that is invading his family. The farmer will join the plants with special abilities to kill the zombies.



Introduction

Game rule

The player controls the plants with special skills into the squares on the lawn to kill the zombies.

Each type of zombie will have a different amount of health

Victory: when killing all zombies of the level.

Defeat: zombies pass through the yard and enter the house

Menu screen



Start new game

**Save player
name**

**Features coming
out in the future**

**Exit game
quickly**

Menu screen



Jlabel

Jtextfield

Jlabel

Jlabel

Menu screen



Code

```
JLabel lblNewLabel = new JLabel("Graph");  
textField_1 = new JTextField();  
textField_1.setFont(new Font("Times New Roman", Font.PLAIN, 11));  
textField_1.setBounds(59, 52, 231, 25);  
lblNewLabel.setBounds(0, 0, 776, 531);  
lblNewLabel.setIcon(new ImageIcon("lib/image/Captureoldmode.png"));  
panel.add(lblNewLabel);
```


Game Background



Code

```
public void setBackground(Graphics g) {  
    backGround = new ImageIcon("lib/image/GameBackground.png").getImage();  
    g.drawImage(backGround, 0, 0, null);  
}
```

Plant



Pea Shooter

Attack on zombie: 15 HP

Cool Down: 1 second

Effect by attack: - 0.01s



Plant



Freeze Pea

Attack on zombie: 25 HP

Cool Down: 0.5 second

Effect by attack: - 0.1s



Plant



Code

```
public static float getStartCoolDown(int plantType) {  
    switch (plantType) {  
        case sunFlower:  
            return 0f;  
        case peaShooter:  
            return 100f;  
        case freezePea:  
            return 50f;  
    }  
    return 0;  
}  
  
public static float getAffectOnZombie(int plantType) {  
    switch (plantType) {  
        case sunFlower:  
            return 0f;  
        case peaShooter:  
            return 10f;  
        case freezePea:  
            return 100f;  
    }  
    return 0;  
}
```


Plant



Code

```
public static float getStartRange(int plantType) {  
    switch (plantType) {  
        case sunFlower:  
            return 0f;  
        case peaShooter:  
            return 1000f;  
        case freezePea:  
            return 1000f;  
    }  
    return 0;  
}
```

Plant



Code

```
public void attack(Plant plant, Zombie zombie) {  
    if (zombie.getAlive()) {  
        if (plant.isCoolDownOver()) {  
            if (checkX(plant, zombie) && checkY(plant, zombie)) {  
                gamePanel.shootZombie(plant, zombie);  
                zombie.setHealth((int) (zombie.getHealth() - plant.getDmg()));  
                int affectZombieSpeed = (int)  
                    model.Helper.Constant.Plants.getAffectOnZombie(plant.getPlantType());  
                zombie.setSpeed(zombie.getSpeed() - affectZombieSpeed);  
                System.out.println(zombie.getHealth());  
                zombie.checkAlive();  
                plant.resetCoolDown();  
            }  
        }  
    }  
}
```


Zombie



Normal zombie

Health: 150 HP
Speed: 4

Zombie



Cone zombie

Health: 250 HP
Speed: 3

Zombie



Boss zombie

Health: 300 HP
Speed: 2

Code

```
public static float getSpeed(int zombieType) {  
    switch (zombieType) {  
        case normalZombie:  
            return -0.25f;  
        case coneZombie:  
            return -0.2f;  
        case finalZombie:  
            return -0.3f;  
        case bossZombie:  
            return -0.14f;  
    }  
    return 0;  
}
```


Zombie



Code

```
public static int getStartHealth(int zombieType) {  
    switch (zombieType) {  
        case normalZombie:  
            return 150;  
        case coneZombie:  
            return 250;  
        case finalZombie:  
            return 300;  
        case bossZombie:  
            return 300;  
    }  
    return 0;  
}
```

Code

```
while (true) {  
  
    now = System.nanoTime();  
    countTime = System.currentTimeMillis();  
    // Render  
    if (now - lastTimeFPS >= timePerFrame) {  
        frame++;  
        lastTimeFPS = System.nanoTime();  
        repaint();  
    }  
  
    // Update  
    if (now - lastTimeUPS >= timePerUpdate) {  
        updateGame++;  
        updateGame();  
        lastTimeUPS = System.nanoTime();  
    }  
  
    // FPS Counter & UPS Counter  
    if (System.currentTimeMillis() - lastTimeCheck >= 1000) {  
        String rs = "FPS: " + frame + "| UPS: " + updateGame + "| Time On Game: "  
            + (int) (countTime - startTime) / 1000 + " s";  
        jLabel.setText(rs);  
        updateGame = reset;  
        frame = reset;  
        lastTimeCheck = System.currentTimeMillis();  
    }  
}
```

FPS counter

UPS counter

Other features



Plants Vs Zombie

FPS: 80 | UPS: 60 | Time On Game: 4 s

**Show the value of FPS
UPS and playing time**

Code

```
public void update() {  
    int zombieKilled = 0;  
    int numZombie = this.zombies.size();  
    for (Zombie zombie : zombies) {  
        if (zombie.getAlive()) {  
            zombie.move(getSpeed(zombie.getZombieType()), y:0f);  
            this.exit(zombie);  
        }  
        if (!zombie.getAlive()) {  
            zombieKilled++;  
        }  
        if (zombieKilled == numZombie) {  
            this.winGame();  
        }  
    }  
}
```

Other features



Effect and rules to
end the game when
killing all zombies

Code

```
public void exit(Zombie zombie) {  
    if (zombie.getX() <= 0) {  
        this.gameOver = new GameOver();  
        gamePanel.getGame().setVisible(false);  
        gamePanel.getGame().getGameTheard().interrupt();  
    }  
}
```

Other features



**Effect and rules to end
the game when zombies
enter the house**

Code

```
public void initButtons() {  
    int xButtonPause = 880;  
    int yButtonPause = y-20;  
    int widthButtonPause = 100;  
    int heightButtonPause = 100;  
    buttonsPause = new MyButton(text:"", xButtonPause, yButtonPause, widthButtonPause, heightButtonPause);  
    plantButtons = new MyButton[3];  
    int x = 10;  
    int y = 10;  
    int addOn = 10;  
    int step = ((int) 1.9f * w) + addOn;  
  
    for (int i = 0; i < plantButtons.length; i++) {  
        plantButtons[i] = new MyButton(text:"", x + (step * i), y, w, h);  
    }  
}  
  
public void draw(Graphics g) {  
    drawButton(g);  
    drawDisplayPlant(g);  
    drawPauseGame(g);  
}
```

Other features



Pause game button



Code

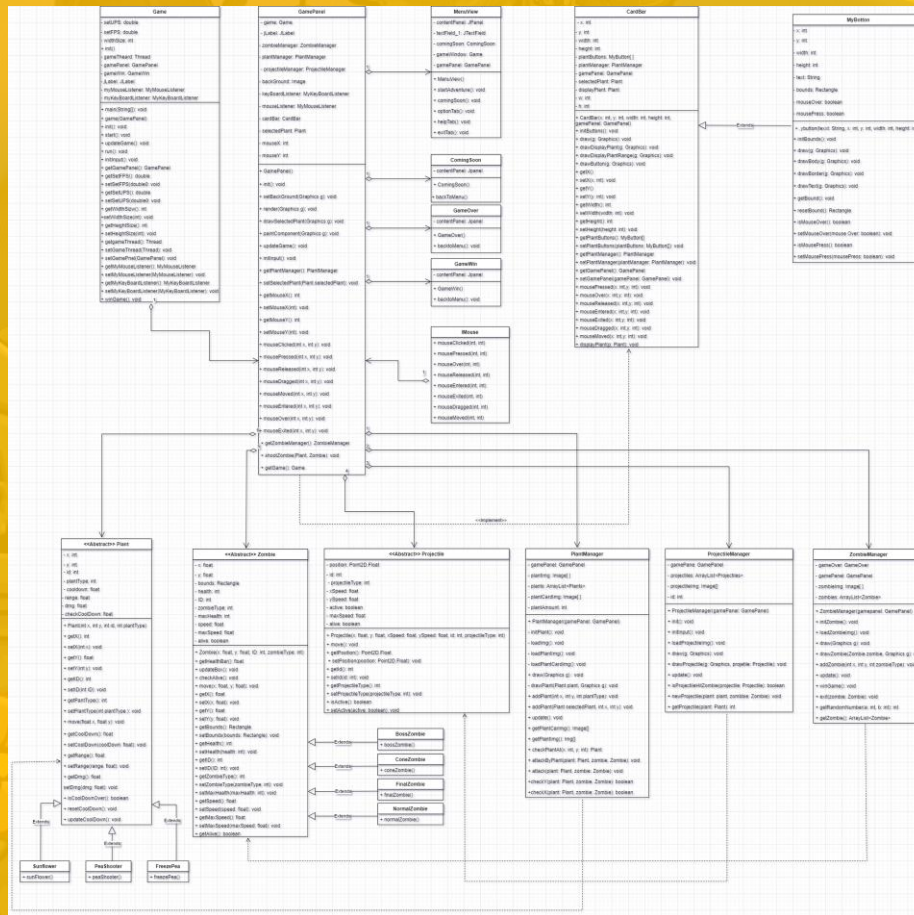
```
public void musicBackground(){  
    String filePath="lib/sound/Grasswalk.wav";  
    try {  
        File file = new File(filePath);  
        if(file.exists()){  
            AudioInputStream ais = AudioSystem.getAudioInputStream(file);  
            Clip clip = AudioSystem.getClip();  
            clip.open(ais);  
            clip.start();  
        }  
    } catch (Exception e) {  
    }  
}
```

Other features



Background Music

UML DIAGRAM





**THANK YOU FOR
YOUR ATTENTION**

CREATED BY: CHICKEN TEAM