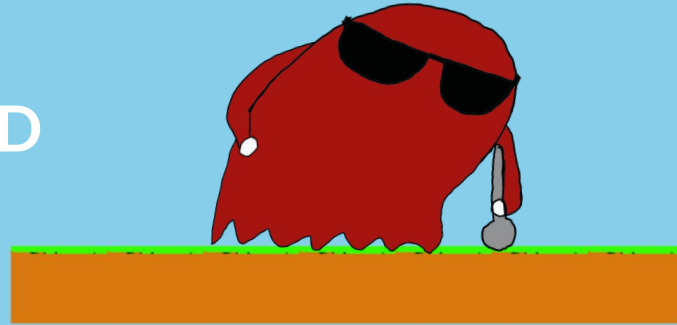


“Frijole Fiasco”, or

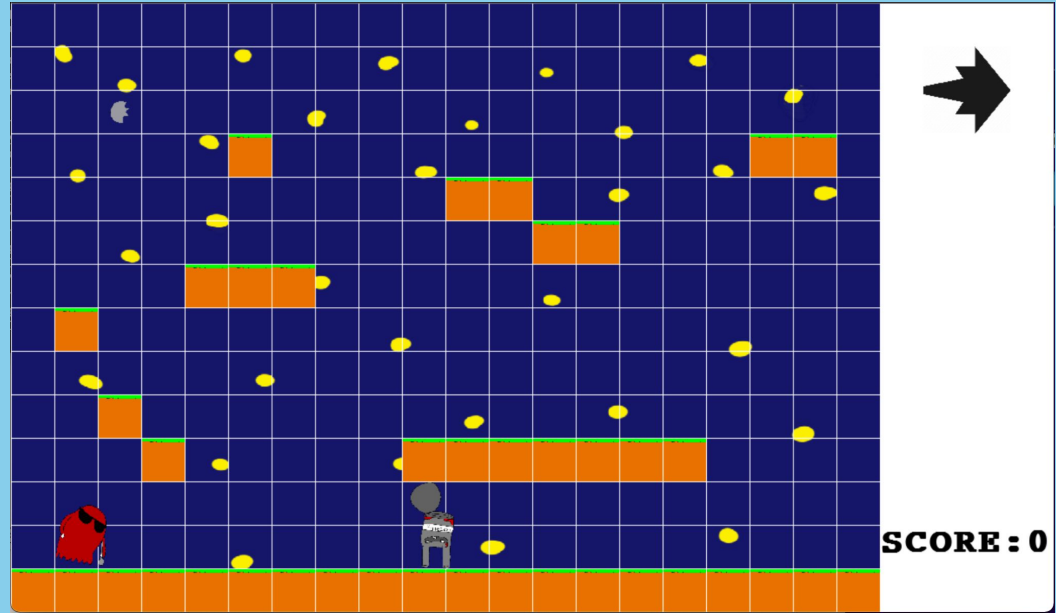
WELCOME TO:
BEAN THING!
(working title)

By: Nolan, Atul, and JD



The Game

Our game is Frijole Fiasco! You play as a half eaten bean in a zombie outbreak where your only movement is dashing left and right. You earn points as you pass to the other side of the screen, while avoiding enemies. At a certain point, the enemies will increase in speed.



The Code

- Classes were created for the game-world, the player, and the enemy.
- Compared to the player class who was given free range of movement, the enemy's movement was set to a jumping pattern and a variable movement speed.
- Made good use of game states, or multiple different game loops for different "screens".
- Atul was able to create a fully programmable game-world with different sky and terrain blocks.
- Comments help to organize and state the use of different code snippets, especially when combining multiple peoples' work.
- Nolan - Working process was to first create mechanics that work, and then carefully simplify and integrate it into where it needed to go (i.e gamestates, classes, etc etc.)

```

// Game World
// This class represents the game world, which contains the player, enemy, and terrain blocks.
// It also handles the game state and the game loop.

class GameWorld {
public:
    // Game state
    int state = 0; // 0: Start screen, 1: Game screen, 2: End screen
    int score = 0;
    int time = 0;
    int level = 0;

    // Game objects
    Player player;
    Enemy enemy;
    Terrain terrain;

    // Game methods
    void start();
    void game();
    void end();

    // Game variables
    float player_x = 0;
    float player_y = 0;
    float enemy_x = 0;
    float enemy_y = 0;
    float terrain_x = 0;
    float terrain_y = 0;
    float score_x = 0;
    float score_y = 0;
    float time_x = 0;
    float time_y = 0;
    float level_x = 0;
    float level_y = 0;
};

// Game World methods
void GameWorld::start() {
    // Start screen
    state = 0;
    score = 0;
    time = 0;
    level = 0;
    player_x = 0;
    player_y = 0;
    enemy_x = 0;
    enemy_y = 0;
    terrain_x = 0;
    terrain_y = 0;
    score_x = 0;
    score_y = 0;
    time_x = 0;
    time_y = 0;
    level_x = 0;
    level_y = 0;
}

void GameWorld::game() {
    // Game screen
    state = 1;
    score = 0;
    time = 0;
    level = 0;
    player_x = 0;
    player_y = 0;
    enemy_x = 0;
    enemy_y = 0;
    terrain_x = 0;
    terrain_y = 0;
    score_x = 0;
    score_y = 0;
    time_x = 0;
    time_y = 0;
    level_x = 0;
    level_y = 0;
}

void GameWorld::end() {
    // End screen
    state = 2;
    score = 0;
    time = 0;
    level = 0;
    player_x = 0;
    player_y = 0;
    enemy_x = 0;
    enemy_y = 0;
    terrain_x = 0;
    terrain_y = 0;
    score_x = 0;
    score_y = 0;
    time_x = 0;
    time_y = 0;
    level_x = 0;
    level_y = 0;
}

// Game World variables
float GameWorld::player_x = 0;
float GameWorld::player_y = 0;
float GameWorld::enemy_x = 0;
float GameWorld::enemy_y = 0;
float GameWorld::terrain_x = 0;
float GameWorld::terrain_y = 0;
float GameWorld::score_x = 0;
float GameWorld::score_y = 0;
float GameWorld::time_x = 0;
float GameWorld::time_y = 0;
float GameWorld::level_x = 0;
float GameWorld::level_y = 0;
```

```

// Game World
// This class represents the game world, which contains the player, enemy, and terrain blocks.
// It also handles the game state and the game loop.

class GameWorld {
public:
    // Game state
    int state = 0; // 0: Start screen, 1: Game screen, 2: End screen
    int score = 0;
    int time = 0;
    int level = 0;

    // Game objects
    Player player;
    Enemy enemy;
    Terrain terrain;

    // Game methods
    void start();
    void game();
    void end();

    // Game variables
    float player_x = 0;
    float player_y = 0;
    float enemy_x = 0;
    float enemy_y = 0;
    float terrain_x = 0;
    float terrain_y = 0;
    float score_x = 0;
    float score_y = 0;
    float time_x = 0;
    float time_y = 0;
    float level_x = 0;
    float level_y = 0;
};

// Game World methods
void GameWorld::start() {
    // Start screen
    state = 0;
    score = 0;
    time = 0;
    level = 0;
    player_x = 0;
    player_y = 0;
    enemy_x = 0;
    enemy_y = 0;
    terrain_x = 0;
    terrain_y = 0;
    score_x = 0;
    score_y = 0;
    time_x = 0;
    time_y = 0;
    level_x = 0;
    level_y = 0;
}

void GameWorld::game() {
    // Game screen
    state = 1;
    score = 0;
    time = 0;
    level = 0;
    player_x = 0;
    player_y = 0;
    enemy_x = 0;
    enemy_y = 0;
    terrain_x = 0;
    terrain_y = 0;
    score_x = 0;
    score_y = 0;
    time_x = 0;
    time_y = 0;
    level_x = 0;
    level_y = 0;
}

void GameWorld::end() {
    // End screen
    state = 2;
    score = 0;
    time = 0;
    level = 0;
    player_x = 0;
    player_y = 0;
    enemy_x = 0;
    enemy_y = 0;
    terrain_x = 0;
    terrain_y = 0;
    score_x = 0;
    score_y = 0;
    time_x = 0;
    time_y = 0;
    level_x = 0;
    level_y = 0;
}

// Game World variables
float GameWorld::player_x = 0;
float GameWorld::player_y = 0;
float GameWorld::enemy_x = 0;
float GameWorld::enemy_y = 0;
float GameWorld::terrain_x = 0;
float GameWorld::terrain_y = 0;
float GameWorld::score_x = 0;
float GameWorld::score_y = 0;
float GameWorld::time_x = 0;
float GameWorld::time_y = 0;
float GameWorld::level_x = 0;
float GameWorld::level_y = 0;
```

Gravity?

```
def update(self):  
    dude.gravity += 1  
    dude.rect.y += dude.gravity          # Natural-ish gravity mechanic.  
  
def jump(self):  
    # if self.rect.bottom == 650:  
    |    self.gravity = -20
```

```
# Right Dash.  
dude.left_inertia -= 1  
if dude.left_inertia < 0:  
    |    dude.left_inertia = 0  
dude.rect.right += dude.left_inertia  
  
# Left Dash.  
dude.right_inertia -= 1  
if dude.right_inertia < 0:  
    |    dude.right_inertia = 0  
dude.rect.left -= dude.right_inertia
```

Our Dynamic

- Nolan and Atul focused on programming, while JD worked on art/graphics for the terrain, background, and players/enemies.
- JD makes all his commits to the “gamegraphics” folder, uploading images and assets for use in the game.
- Atul and Nolan worked on different files and would periodically combine their work.
- Nolan worked on the player code and the movement mechanics. He also worked on the enemy. Atul primarily worked on the background and collision mechanics.

gamegraphics	jumping enemy
GameStateUpdate_11.29.pdf	uploading milestone update for Tuestay 11/29
README.md	Create README.md
background.py	Update background.py
background2.py	fixed some errors with atul
background3.py	Update background3.py
maingame.py	added start screen
proposal.pdf	adding proposal.pdf
test.py	Update test.py



Our Schedule

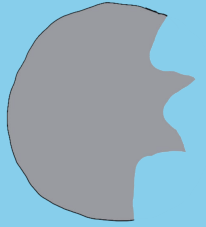
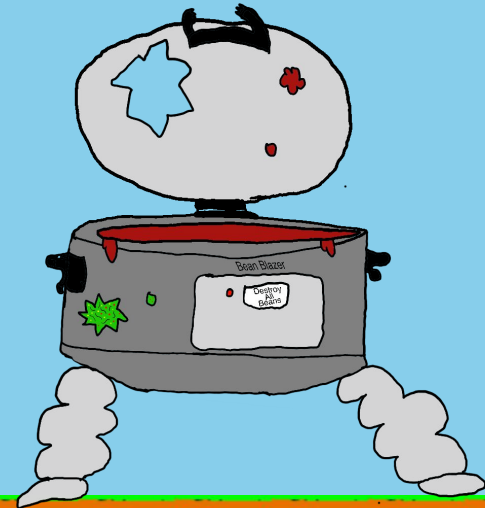
We stayed on schedule for the most part. After thanksgiving, we realized how much time we had left, and we started to move a little faster. We met up after class sometimes to discuss what needed to get done. JD had finished the final graphic for the game within the second to last week of development. Nolan had optimized the resolution after testing it on JD's computer we found out it was too big for smaller computer screens around the same time as well.

There are still a lot of things that we want to add that we couldn't because other things aren't working. We wanted to add more enemies (JD drew a broccoli with realistic human legs) and more levels, but we couldn't get the existing ones to work.



Future Ideas

- More enemies
- Sound/Background Music
- Making the world bigger so that we can
add more enemies and a final boss →



Key Takeaways

We learned to never complain about game dev again and we now understand what it takes to develop a game.

Atul:

- I no longer want to be a game developer
- I hate pygame
- It would've been a lot more fun if everything worked as planned, but it really helped open my eyes to all the possibilities through programming (corny but true)
- Made me more interested in programming (but not pygame or related things)

Nolan:

- Learned the importance of organization in code
- Underestimated effort that goes into simple games
- Became uncomfortably familiar with programming classes
- MERGE. CONFLICTS. COMMUNICATION.

JD:

- I am much better at graphics than coding
- DON'T FORGET TO PUSH GRAPHICS BEFORE PULLING SO WE DON'T THINK THE GAME BROKE
- Made me want to learn more how to make a game
- Anti-cheat

Thank You!



Saint Picklous Today at 2:22 PM

But what do you think of bean tho
Is he hot



p1 Today at 2:22 PM

Yes



jolly hung eater Today at 2:30 PM

One of the games I've played ever



TWCO Viz Today at 2:30 PM

ykw nvm



BWST sexier Today at 2:30 PM

gave me meningitis :(



(sprites used in beta)

