Nodify

Group 7 - Capstone Proof of Concept

Problem Statement

- Lacklustre documentation makes contributions and maintenance difficult
- Understanding code with insufficient documentation wastes time
- Reading source code to understand a program reduces programming productivity
- Visualization of large codebases is extremely difficult
  - Learning
  - Performance monitoring

Objectives

- → Generate interactive execution flow diagrams for a codebase
- → Abstract the flow of execution for maximum understandability
- → Interact with the program through static analysis
- → Catch performance bottlenecks through visualization

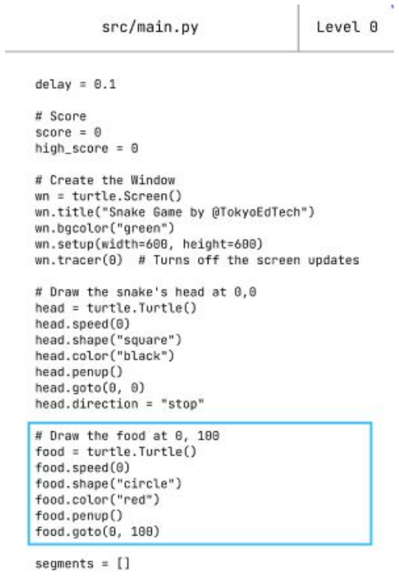
Stakeholders

- Software engineers
  - Especially to ramp up new team members
- Programming project contributors
- Mentors, teachers
- DevOps, QA, Performance Engineers

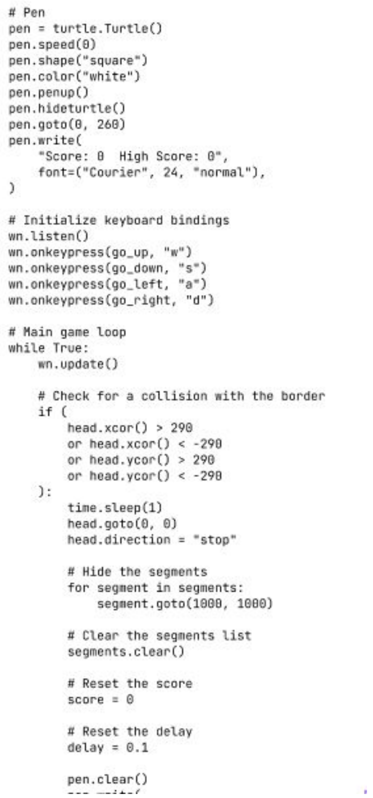
Solution - Nodify

- → Interactive flow-diagram generator to visualize programs within an IDE
- → Summarize and abstract complex details into understandable components
- → Collect and display performance metrics and telemetry data

Plain Code

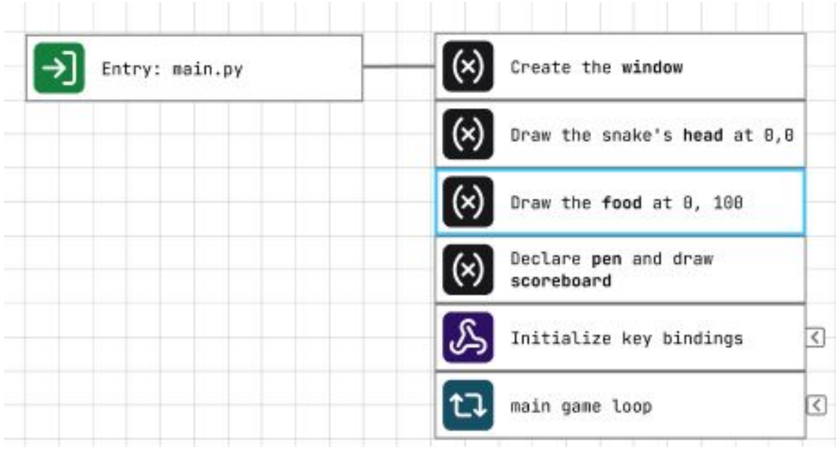


```
src/main.py                Level 0

delay = 0.1

# Score
score = 0
high_score = 0

# Create the Window
wn = turtle.Screen()
wn.title("Snake Game by @TokyoEdTech")
wn.bgcolor("green")
wn.setup(width=600, height=600)
wn.tracer(0)  # Turns off the screen updates

# Draw the snake's head at 0,0
head = turtle.Turtle()
head.speed(0)
head.shape("square")
head.color("black")
head.penup()
head.goto(0, 0)
head.direction = "stop"

# Draw the food at 0, 100
food = turtle.Turtle()
food.speed(0)
food.shape("circle")
food.color("red")
food.penup()
food.goto(0, 100)

segments = []
```



```
# Pen
pen = turtle.Turtle()
pen.speed(0)
pen.shape("square")
pen.color("white")
pen.penup()
pen.hideturtle()
pen.goto(0, 260)
pen.write(
    "Score: 0  High Score: 0",
    font=("Courier", 24, "normal"),
)

# Initialize keyboard bindings
wn.listen()
wn.onkeypress(go_up, "w")
wn.onkeypress(go_down, "s")
wn.onkeypress(go_left, "a")
wn.onkeypress(go_right, "d")

# Main game loop
while True:
    wn.update()

    # Check for a collision with the border
    if (
        head.xcor() > 290
        or head.xcor() < -290
        or head.ycor() > 290
        or head.ycor() < -290
    ):
        time.sleep(1)
        head.goto(0, 0)
        head.direction = "stop"

        # Hide the segments
        for segment in segments:
            segment.goto(1000, 1000)

        # Clear the segments list
        segments.clear()

        # Reset the score
        score = 0

        # Reset the delay
        delay = 0.1

        pen.clear()
```

Level 1 Abstraction



Entry: main.py
Create the window
Draw the snake's head at 0,0
Draw the food at 0, 100
Declare pen and draw scoreboard
Initialize key bindings
main game loop