

Text

At this point, the game texts are shown, and update as needed.

Run.py

```
import pygame
from pygame.locals import *
from constants import *
from pacman import Pacman
from nodes import NodeGroup
from pellets import PelletGroup
from ghosts import GhostGroup
from fruit import Fruit
from pauser import Pause
from text import TextGroup

class GameController(object):
    def __init__(self):
        pygame.init()
        self.screen = pygame.display.set_mode(SCREENSIZE, 0, 32)
        self.background = None
        self.clock = pygame.time.Clock()
        self.fruit = None
        self.pause = Pause(True)
        self.level = 0
        self.lives = 5
        self.score = 0
        self.textgroup = TextGroup()

    def setBackground(self):
        self.background = pygame.surface.Surface(SCREENSIZE).convert()
        self.background.fill(BLACK)

    def startGame(self):
        self.setBackground()
        self.nodes = NodeGroup("maze01.txt")
        self.nodes.setPortalPair((0,17), (27,17))
        homekey = self.nodes.createHomeNodes(11.5, 14)
        self.nodes.connectHomeNodes(homekey, (12,14), LEFT)
        self.nodes.connectHomeNodes(homekey, (15,14), RIGHT)
        self.pacman = Pacman(self.nodes.getNodeFromTiles(15, 26))
        self.pellets = PelletGroup("maze01.txt.")
        self.ghosts = GhostGroup(self.nodes.getStartTempNode(), self.pacman)
        self.ghosts.blinky.setStartNode(self.nodes.getNodeFromTiles(2+11.5, 0+14))
        self.ghosts.pinky.setStartNode(self.nodes.getNodeFromTiles(2+11.5, 3+14))
        self.ghosts.inky.setStartNode(self.nodes.getNodeFromTiles(0+11.5, 3+14))
        self.ghosts.clyde.setStartNode(self.nodes.getNodeFromTiles(4+11.5, 3+14))
```

```

self.ghosts.setSpawnNode(self.nodes.getNodeFromTiles(2+11.5, 3+14))
self.nodes.denyHomeAccess(self.pacman)
self.nodes.denyHomeAccessList(self.ghosts)
self.nodes.denyAccessList(2+11.5, 3+14, LEFT, self.ghosts)
self.nodes.denyAccessList(2+11.5, 3+14, RIGHT, self.ghosts)
self.ghosts.inky.startNode.denyAccess(RIGHT, self.ghosts.inky)
self.ghosts.clyde.startNode.denyAccess(LEFT, self.ghosts.clyde)
self.nodes.denyAccessList(12, 14, UP, self.ghosts)
self.nodes.denyAccessList(15, 14, UP, self.ghosts)
self.nodes.denyAccessList(12, 26, UP, self.ghosts)
self.nodes.denyAccessList(15, 26, UP, self.ghosts)

def update(self):
    dt = self.clock.tick(30) / 1000.0
    self.textgroup.update(dt)
    self.pellets.update(dt)
    if not self.pause.paused:
        self.pacman.update(dt)
        self.ghosts.update(dt)
        if self.fruit is not None:
            self.fruit.update(dt)
        self.checkGhostEvents()
        self.checkPelletEvents()
        self.checkFruitEvents
    afterPauseMethod = self.pause.update(dt)
    if afterPauseMethod is not None:
        afterPauseMethod()
    self.checkEvents()
    self.render()

def updateScore(self, points):
    self.score += points
    self.textgroup.updateScore(self.score)

def checkEvents(self):
    for event in pygame.event.get():
        if event.type == QUIT:
            exit()
        elif event.type == KEYDOWN:
            if event.key == K_SPACE:
                if self.pacman.alive:
                    self.pause.setPause(playerPaused=True)
                    if not self.pause.paused:
                        self.textgroup.hideText()
                        self.showEntities()
                else:
                    self.hideEntities()
                    self.textgroup.showText(PAUSETXT)

def checkGhostEvents(self):
    for ghost in self.ghosts:
        if self.pacman.collideGhost(ghost):
            if ghost.mode.current is FREIGHT:

```

```

        self.pacman.visible = False
        ghost.visible = False
        self.updateScore(ghost.points)
        self.textgroup.addText(str(ghost.points), WHITE, ghost.position.x, ghost.position.y, 8, time=1)
        self.ghosts.updatePoints()
        self.pause.setPause(pauseTime=1, func=self.showEntities)
        ghost.startSpawn()
        self.nodes.allowHomeAccess(ghost)
    elif ghost.mode.current is not SPAWN:
        if self.pacman.alive:
            self.lives -= 1
            self.pacman.die()
            self.ghosts.hide()
            if self.lives <= 0:
                self.textgroup.showText(GAMEOVERTXT)
                self.pause.setPause(pauseTime=3, func=self.restartGame)
            else:
                self.pause.setPause(pauseTime=3, func=self.resetLevel)

def checkPelletEvents(self):
    pellet = self.pacman.eatPellets(self.pellets.pelletList)
    if pellet:
        self.pellets.numEaten += 1
        self.updateScore(pellet.points)
        if self.pellets.numEaten == 30:
            self.ghosts.inky.startNode.allowAccess(RIGHT, self.ghosts.inky)
        if self.pellets.numEaten == 70:
            self.ghosts.clyde.startNode.allowAccess(LEFT, self.ghosts.clyde)
        self.pellets.pelletList.remove(pellet)
        if pellet.name == POWERPELLET:
            self.ghosts.startFreight()
        if self.pellets.isEmpty():
            self.hideEntities()
            self.pause.setPause(pauseTime=3, func=self.nextLevel)

def checkFruitEvents(self):
    if self.pellets.numEaten == 50 or self.pellets.numEaten == 140:
        if self.fruit is None:
            self.fruit = Fruit(self.nodes.getNodeFromTiles(9, 20))
        if self.fruit is not None:
            if self.pacman.collideCheck(self.fruit):
                self.updateScore(self.fruit.points)
                self.textgroup.addText(str(self.fruit.points), WHITE, self.fruit.position.x, self.fruit.position.y, 8,
time=1)
            self.fruit = None
        elif self.fruit.destroy:
            self.fruit = None

def showEntities(self):
    self.pacman.visible = True
    self.ghosts.show()

def hideEntities(self):

```

```

        self.pacman.visible = False
        self.ghosts.hide()

    def nextLevel(self):
        self.showEntities()
        self.level += 1
        self.pause.paused = True
        self.textgroup.updateLevel(self.level)
        self.startGame()

    def restartGame(self):
        self.lives = 5
        self.level = 0
        self.pause.paused = True
        self.fruit = None
        self.score = 0
        self.textgroup.updateScore(self.score)
        self.textgroup.updateLevel(self.level)
        self.textgroup.showText(READYTXT)
        self.startGame()

    def resetLevel(self):
        self.pause.paused = True
        self.pacman.reset()
        self.ghosts.reset()
        self.fruit = None
        self.texrgroup.showText(READYTXT)

    def render(self):
        self.screen.blit(self.background, (0,0))
        self.nodes.render(self.screen)
        self.pellets.render(self.screen)
        if self.fruit is not None:
            self.fruit.render(self.screen)
        self.pacman.render(self.screen)
        self.ghosts.render(self.screen)
        self.textgroup.render(self.screen)
        pygame.display.update()

if __name__ == "__main__":
    game = GameController()
    game.startGame()
    while True:
        game.update()

```

Text.py

```

import pygame
from vector import Vector2
from constants import *

```

```

class Text(object):
    def __init__(self, text, color, x, y, size, time=None, id=None, visible=True):
        self.id = id
        self.text = text
        self.color = color
        self.size = size
        self.visible = visible
        self.position = Vector2(x, y)
        self.timer = 0
        self.lifespan = time
        self.label = None
        self.destroy = False
        self.setupFont("PressStart2P-Regular.ttf")
        self.createLabel()

    def setupFont(self, fontpath):
        self.font = pygame.font.Font(fontpath, self.size)

    def createLabel(self):
        self.label = self.font.render(self.text, 1, self.color)

    def setText(self, newtext):
        self.text = str(newtext)
        self.createLabel()

    def update(self, dt):
        if self.lifespan is not None:
            self.timer += dt
            if self.timer >= self.lifespan:
                self.timer = 0
                self.lifespan = None
                self.destroy = True

    def render(self, screen):
        if self.visible:
            x, y = self.position.asTuple()
            screen.blit(self.label, (x, y))

class TextGroup(object):
    def __init__(self):
        self.nextid = 10
        self.alltext = {}
        self.setupText()
        self.showText(READYTXT)

    def addText(self, text, color, x, y, size, time=None, id=None):
        self.nextid += 1
        self.alltext[self.nextid] = Text(text, color, x, y, size, time=time, id=id)
        return self.nextid

    def removeText(self, id):
        self.alltext.pop(id)

```

```

def setupText(self):
    size = TILEHEIGHT
    self.alltext[SCORETXT] = Text("0".zfill(8), WHITE, 0, TILEHEIGHT, size)
    self.alltext[LEVELTXT] = Text(str(1).zfill(3), WHITE, 23*TILEWIDTH, TILEHEIGHT, size)
    self.alltext[READYTXT] = Text("READY!", YELLOW, 11.25*TILEWIDTH, 20*TILEHEIGHT, size,
visible=False)
    self.alltext[PAUSETXT] = Text("PAUSED!", YELLOW, 10.625*TILEWIDTH, 20*TILEHEIGHT, size,
visible=False)
    self.alltext[GAMEOVERTXT] = Text("GAMEOVER!", YELLOW, 10*TILEWIDTH, 20*TILEHEIGHT, size,
visible=False)
    self.addText("SCORE", WHITE, 0, 0, size)
    self.addText("LEVEL", WHITE, 23*TILEWIDTH, 0, size)

def update(self, dt):
    for tkey in list(self.alltext.keys()):
        self.alltext[tkey].update(dt)
        if self.alltext[tkey].destroy:
            self.removeText(tkey)

def showText(self, id):
    self.hideText()
    self.alltext[id].visible = True

def hideText(self):
    self.alltext[READYTXT].visible = False
    self.alltext[PAUSETXT].visible = False
    self.alltext[GAMEOVERTXT].visible = False

def updateScore(self, score):
    self.updateText(SCORETXT, str(score).zfill(8))

def updateLevel(self, level):
    self.updateText(LEVELTXT, str(level + 1).zfill(3))

def updateText(self, id, value):
    if id in self.alltext.keys():
        self.alltext[id].setText(value)

def render(self, screen):
    for tkey in list(self.alltext.keys()):
        self.alltext[tkey].render(screen)

```

Constants.py

```

TILEWIDTH = 16
TILEHEIGHT = 16
NROWS = 36
NCOLS = 28
SCREENWIDTH = NCOLS*TILEWIDTH
SCREENHEIGHT = NROWS*TILEHEIGHT
SCREENSIZE = (SCREENWIDTH, SCREENHEIGHT)

```

BLACK = (0, 0, 0)
GREEN = (0, 255, 0)

YELLOW = (255, 255, 0)

STOP = 0
UP = 1
DOWN = -1
LEFT = 2
RIGHT = -2

PACMAN = 0
PELLET = 1
POWERPELLET = 2
GHOST = 3
BLINKY = 4
PINKY = 5
INKY = 6
CLYDE = 7
FRUIT = 8

WHITE = (255, 255, 255)
RED = (255, 0, 0)

PORTAL = 3

SCATTER = 0
CHASE = 1
FREIGHT = 2
SPAWN = 3

PINK = (255, 100, 150)
TEAL = (100, 255, 255)
ORANGE = (230, 190, 40)

SCORETXT = 0
LEVELTXT = 1
READYTXT = 2
PAUSETXT = 3
GAMEOVERTXT = 4