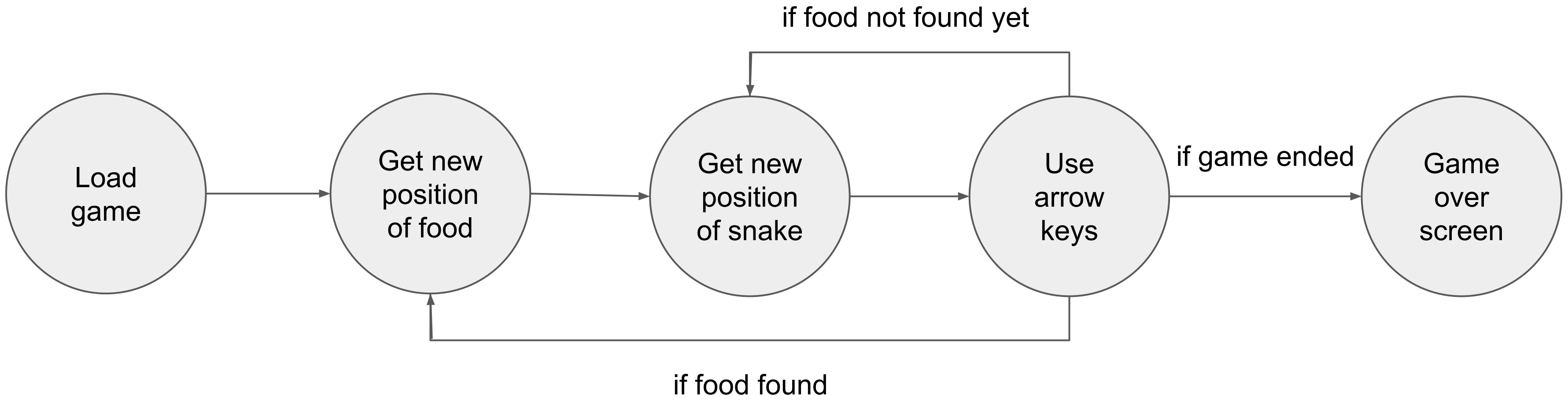
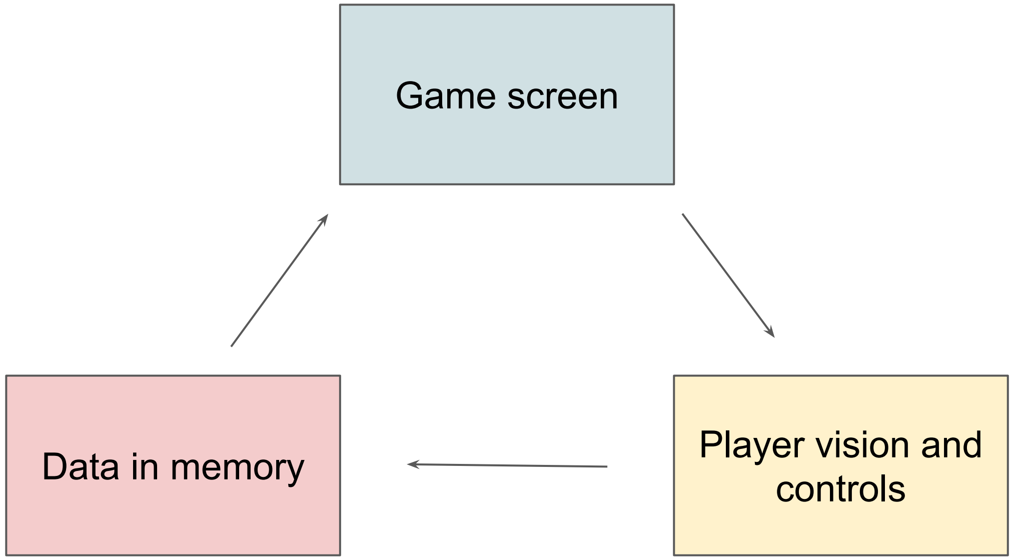
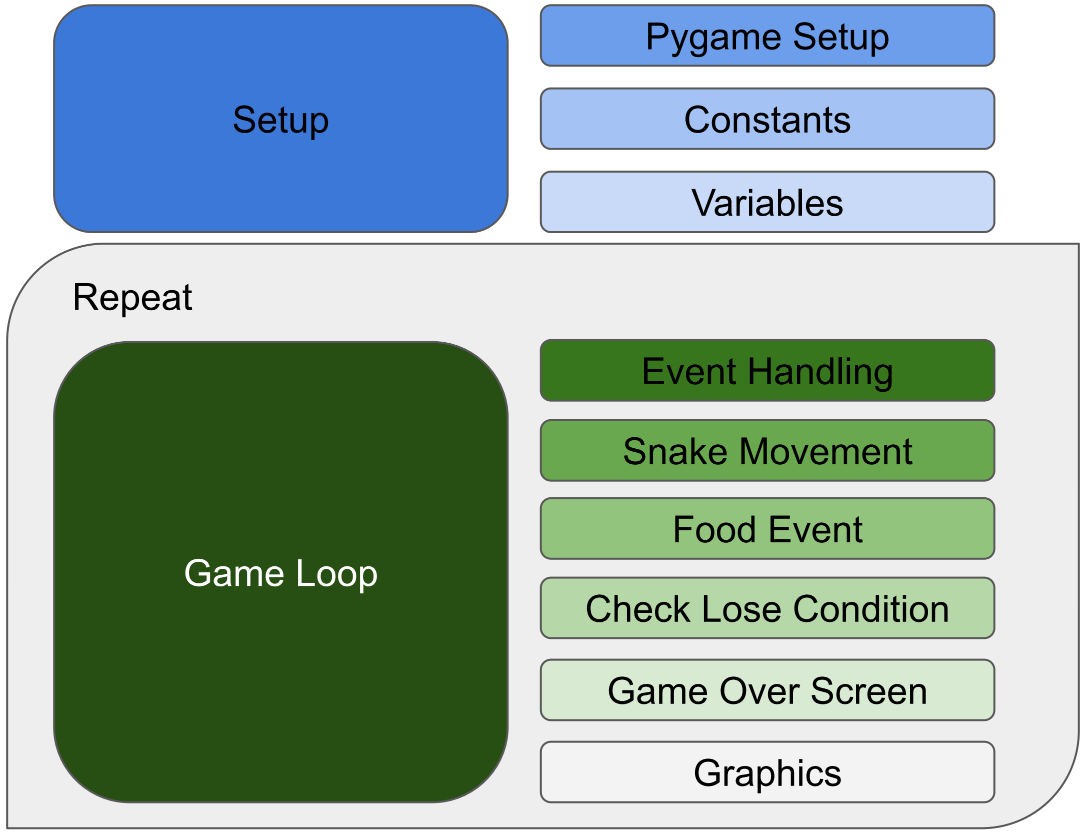
Schematic View of Snake



Communication and Accessibility



Implementation Design

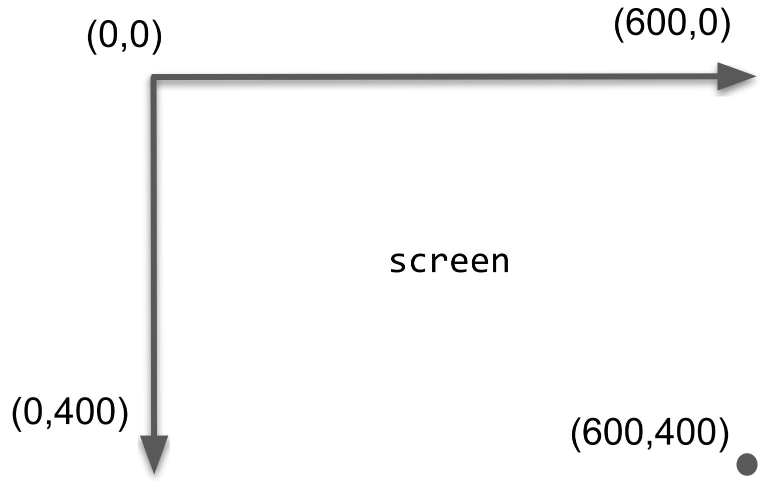


Setup & Break Time! (~5 minutes)

- If you **do not have** Python installed:
  - Go to replit.com and sign up/log in
  - Click “+ Create Repl” on the top left and select Python
- If you **have** Python installed:
  - Ensure that you are using Python 3
  - Install Pygame using pip/Anaconda (e.g. `pip install pygame`) in terminal/command prompt

To access workshop material for this segment, go to github.com/wmloh/workshop

Pygame Graphics



`set_mode((SIZE_X, SIZE_Y))` – create the main Surface object with size `(SIZE_X, SIZE_Y)`

`set_caption(TEXT)` - set window title to `TEXT`

`font.render(TEXT, True, COLOR)` - generate a Surface from `TEXT` with colour `COLOR`

`screen.fill(COLOR)` - fill the main Surface with the colour `COLOR`

`screen.blit(SURF,(X, Y))` - pastes `SURF` onto the main Surface at `(X, Y)`

`update()` - refresh the entire window with latest elements

`draw.rect(SURF, COLOR, (X, Y, SIZE_X, SIZE_Y))` - draw a `COLOR` rectangle on `SURF` at `(X, Y)` with size `(SIZE_X, SIZE_Y)`

Snake Attributes

- Leading coordinate (head)
- Body coordinates
- Body length
- Current direction and velocity

★ Body length grows by 1 after consuming food

★ Body coordinates are previous leading coordinates

Food Attribute

- Coordinates

★ Randomly generated within the bounds of the screen

Snake Movement Process

Assuming that `snake_maxlen` starts with 1

- [t = 1] `(x, y) = (50, 50)`

eats food (`snake_maxlen = 2`)

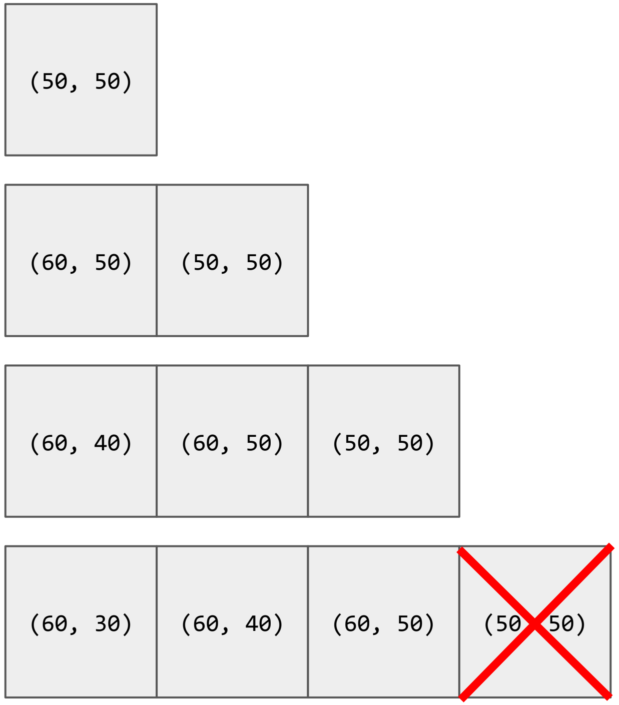
- [t = 2] `(x, y) = (60, 50)`

eats food (`snake_maxlen = 3`)

- [t = 3] `(x, y) = (60, 40)`

no food (`snake_maxlen = 3`)

- [t = 4] `(x, y) = (60, 30)`



Other Game Development Tools

- Python Pyglet
- Java libGDX
- Unity
- Unreal Engine

Syntax Glossary

Expression	Purpose
`break`	Stops execution of the closest for loop or while loop
`clock.tick(15)`	Keeps loop frequency to 15 loops per second
`event.type`	Type of event (e.g. mouse or keyboard)
`event.key`	Type of keyboard event (e.g. K_c, K_q)
`font.render(“hello”, True, BLACK)`	Creates a surface object with black “hello” text
`import pygame`	Imports the library with the name pygame
`len(snake_pos)`	Gets length of the snake_pos list
`list()`	Create a new list object
`pygame.init()`	Imports all core Pygame subpackages

Syntax Glossary

`pygame.display.set_caption(“hello”)`	Sets window title to “hello”
`pygame.display.set_mode((600, 400))`	Create the main surface object with size (600, 400)
`pygame.display.update()`	Refreshes the entire window with latest elements
`pygame.draw.rect(screen,RED,(0,0,10,20))`	Draws a red rectangle on screen at (0, 0) with size (10, 20)
`pygame.event.get()`	Gets all events collected within a certain time frame
`pygame.font.SysFont(“tahoma”, 25)`	Creates a font generator with style “tahoma” of size 25
`pygame.K_LEFT`	Pygame indicator for keyboard left arrow key
`pygame.time.Clock()`	Creates a Clock object
`pygame.quit()`	Terminates and closes Pygame window
`quit()`	Terminates Python execution

Syntax Glossary

`random.randrange(0, 9)`	Gets a random integer between 0 (inclusive) to 9 (exclusive)
`round(314.15926, -1)`	Rounds 314.15926 to the tens place, i.e. yields 310
`screen.fill(GREEN)`	Fill screen surface object with green
`screen.blit(GAME_OVER_TEXT, (20, 20))`	Pastes GAME_OVER_TEXT onto screen at (20, 20)
`snake_pos.insert(0, (x, y))`	Inserts (x, y) at index 0 (i.e. leftmost) of the snake_pos list
`snake_pos.pop()`	Removes the last element (rightmost) of the snake_pos list
`str(snake_maxlen)`	Convert the integer snake_maxlen to a string
`(x, y) in snake_pos[1:]`	Returns True if any index of snake_pos excluding the leftmost index contains (x, y), and False otherwise