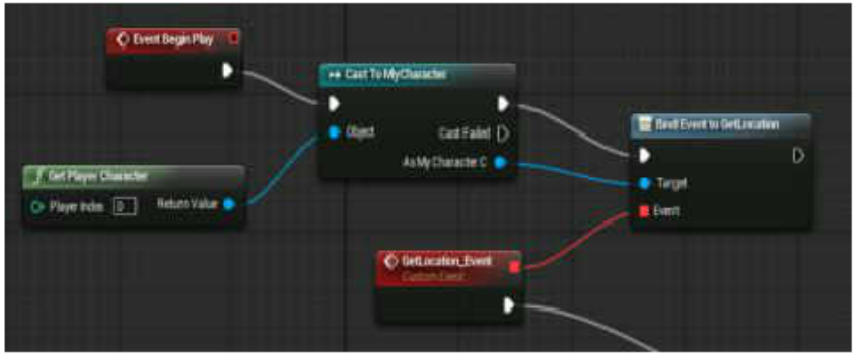
Basics of video game programming

Gaëtan Blaise-Cazalet - Head of programming section

What are we talking about?

~~Visual Scripting~~



Script / Code

```
string sInput;
int iLength, iN;
double dblTemp;
bool again = true;

while (again) {
    iN = -1;
    again = false;
    getline(cin, sInput);
    system("cls");
    stringstream(sInput) >> dblTemp;
    iLength = sInput.length();
    if (iLength < 4) {
        again = true;
        continue;
    } else if (sInput[iLength - 3] != '.') {
        again = true;
        continue;
    } while (++iN < iLength) {
        if (isdigit(sInput[iN])) {
            continue;
        } else if (iN >= (iLength - 3) ) {
```

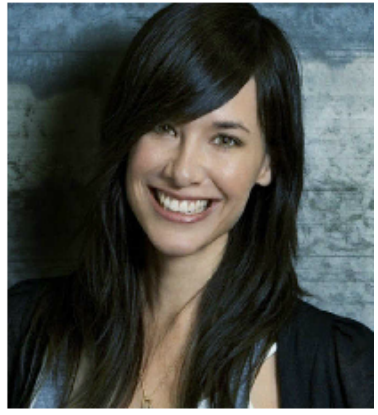
Not so much differences. When you are visual scripting, you are coding :

- In a less efficient way (-)
- Without learning a programming language (+)

Why coding ?

- You make video game. The matter you use, the feeling of your games, is computer code.
- To discuss with him or her

(without being a fool)



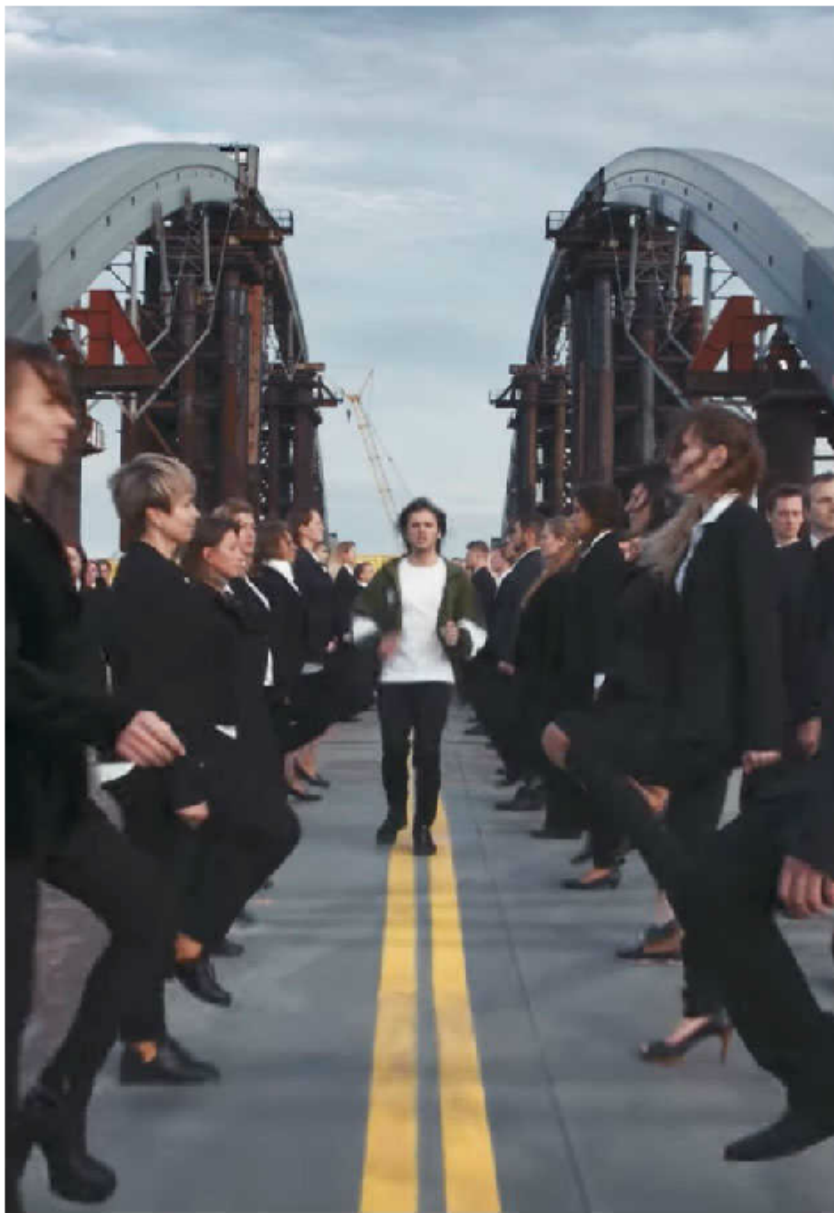
- To prototype your gameplays without depending of anyone

Genesis 1-1 : a basic interaction

Basics of a video game

```
window {
    load();

    loop {
        inputs();
        update();
        draw();
    };
}
```



Talking to your machine

We will use pygame and the python language

- Create a Basics folder
- Open Visual Studio Code
- Open your folder with visual studio code
- Create a main.py file

```
print("hello world")
```

- Hit F5. Click on the cog to add a launch configuration.
- Your program will launch in the terminal.

Main function

- Allow python to know this is the starting point of a program

```
def main():
    print("hello world")

if __name__ == "__main__":
    main()
```

- Indented code is "inside" the former block
- Same feature : print "hello world". This structure will be useful later.

Create a window

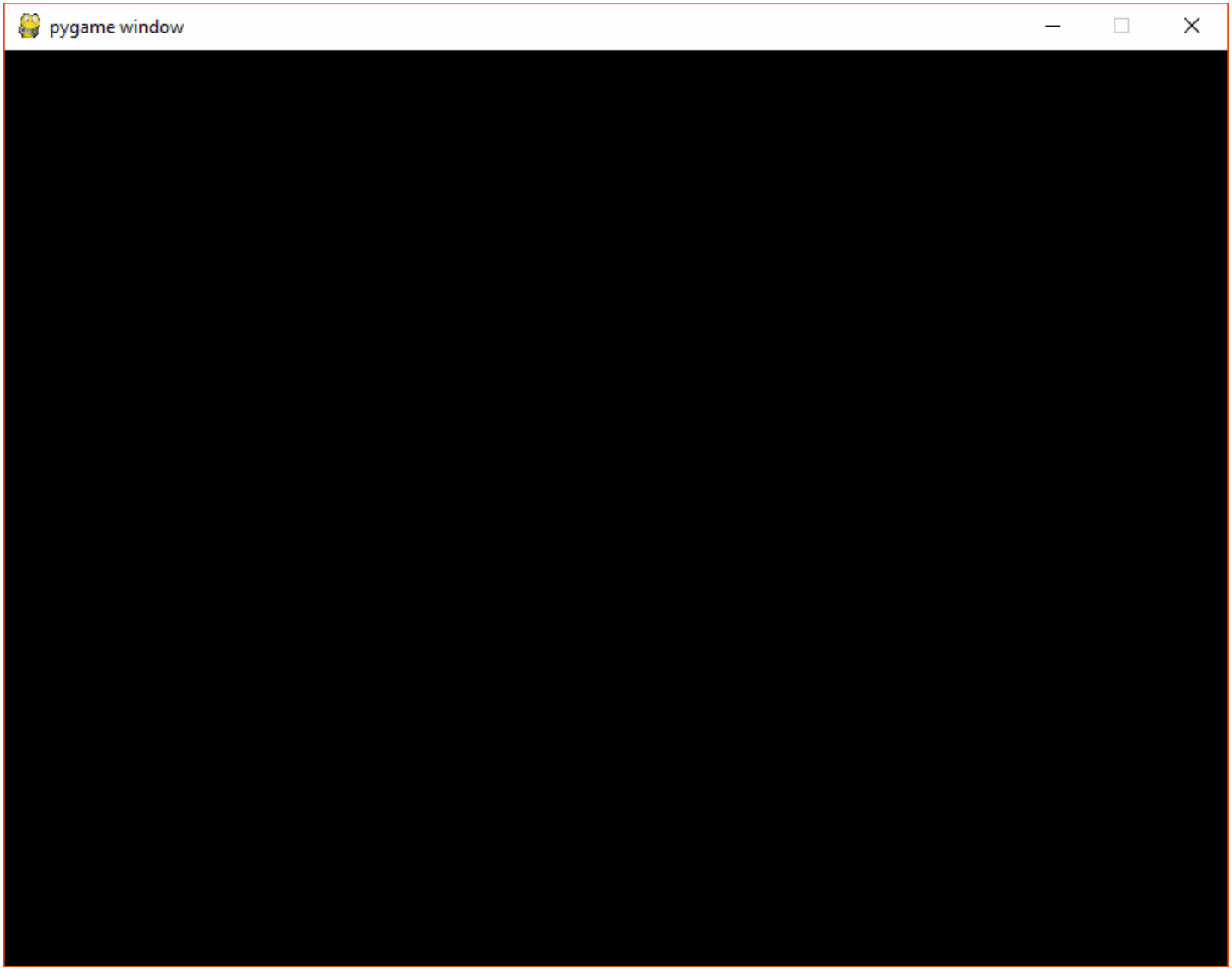
Invoke pygame window

```
import pygame

def main():
    pygame.init()
    screen = pygame.display.set_mode((800, 600))
    # End program


...
```

- "import" call a module to be used in the code.
- pygame.init() calls the init function of he pygame module
- "screen = " means we attribute what is on the right part to a variable called screen
- We create a 800 * 600 window
- Code starting with # is a comment : not interpreted by the machine


pygame window

Draw

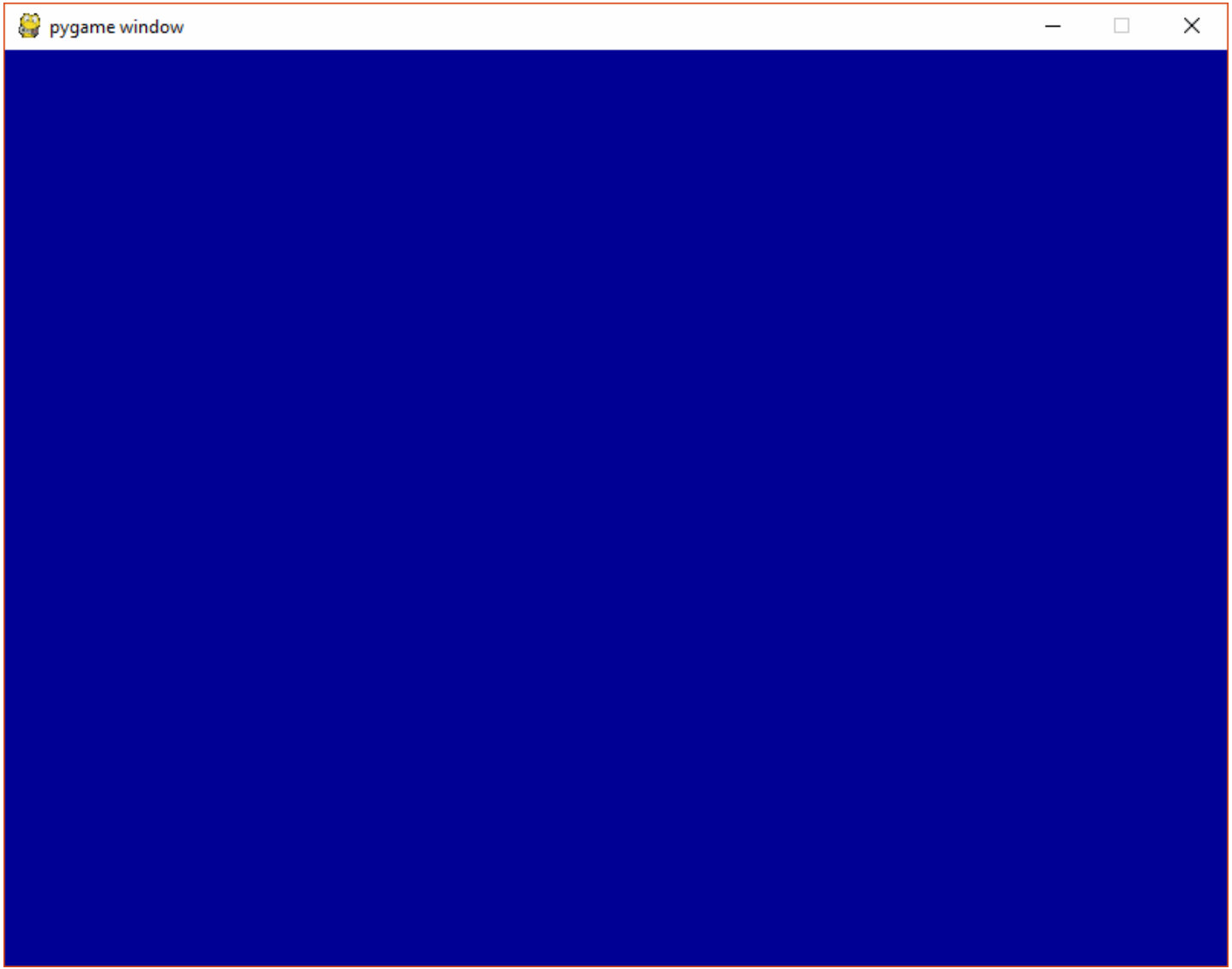
Color in window

- main.py

```
import pygame, time

def main():
    pygame.init()
    screen = pygame.display.set_mode((800, 600))
    screen.fill((0, 0, 150))
    pygame.display.update()
    time.sleep(2) # End program after 2 seconds
```

- Colors are coded with a tuple : 3 values between 0 and 255, representing a melt of Red, Green and Blue


pygame window

Display text

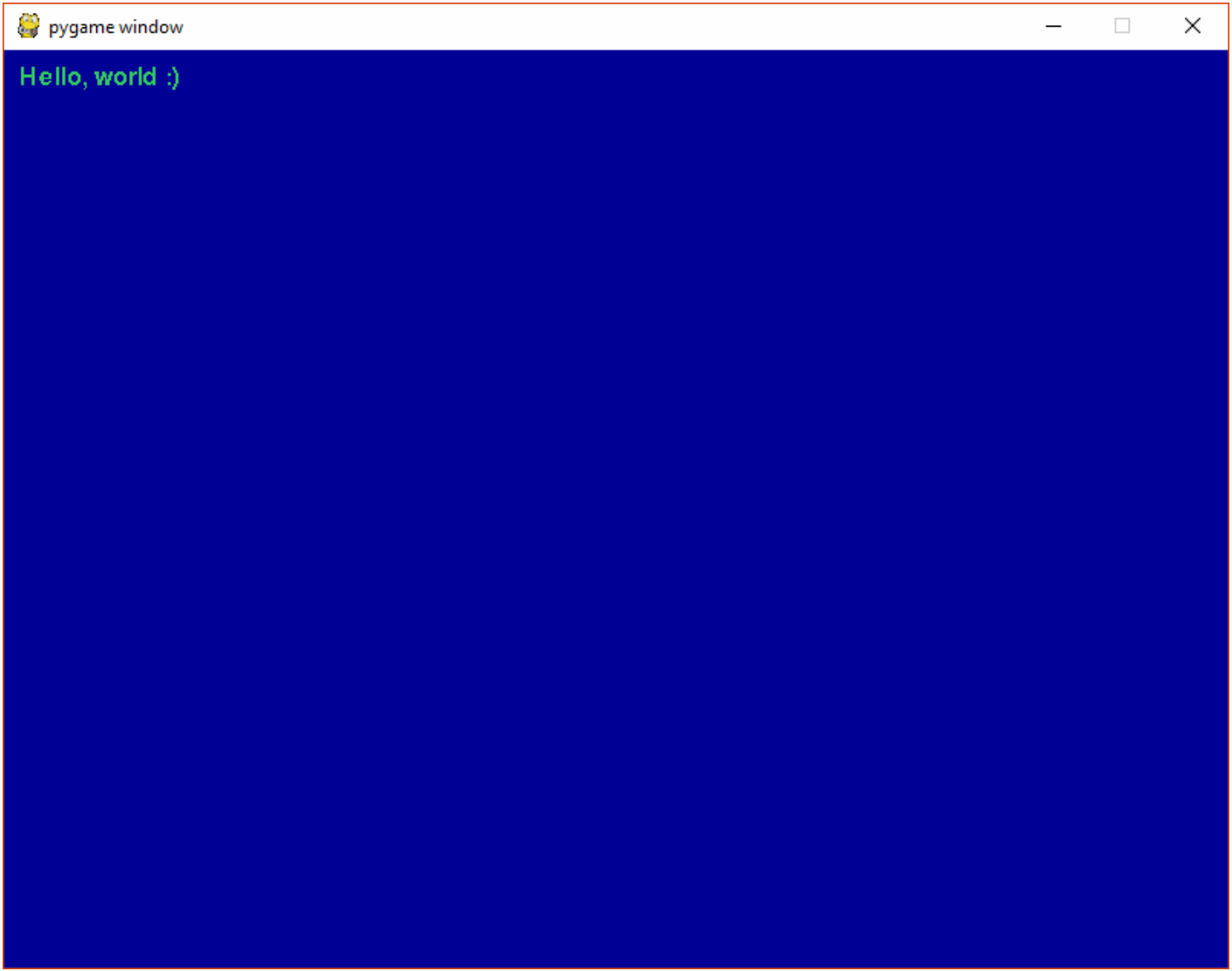
- main.py

```
def main():
    pygame.init()
    screen = pygame.display.set_mode((800, 600))
    screen.fill((0, 0, 150))

    font = pygame.font.Font(None, 24)
    text = font.render("Hello, world :)", False, (50, 200, 100))
    screen.blit(text, (10, 10))

    pygame.display.update()
    time.sleep(2) # End program after 2 seconds
```

- To display a text, you have to load a font (here default font is used) then to write with this font
- blit function add pixels on a surface. Here we add text pixels on screen surface.
- pygame.display.update() switch the current buffer (double buffering)


pygame window
Hello, world :)

Inputs

Change text when hit a key

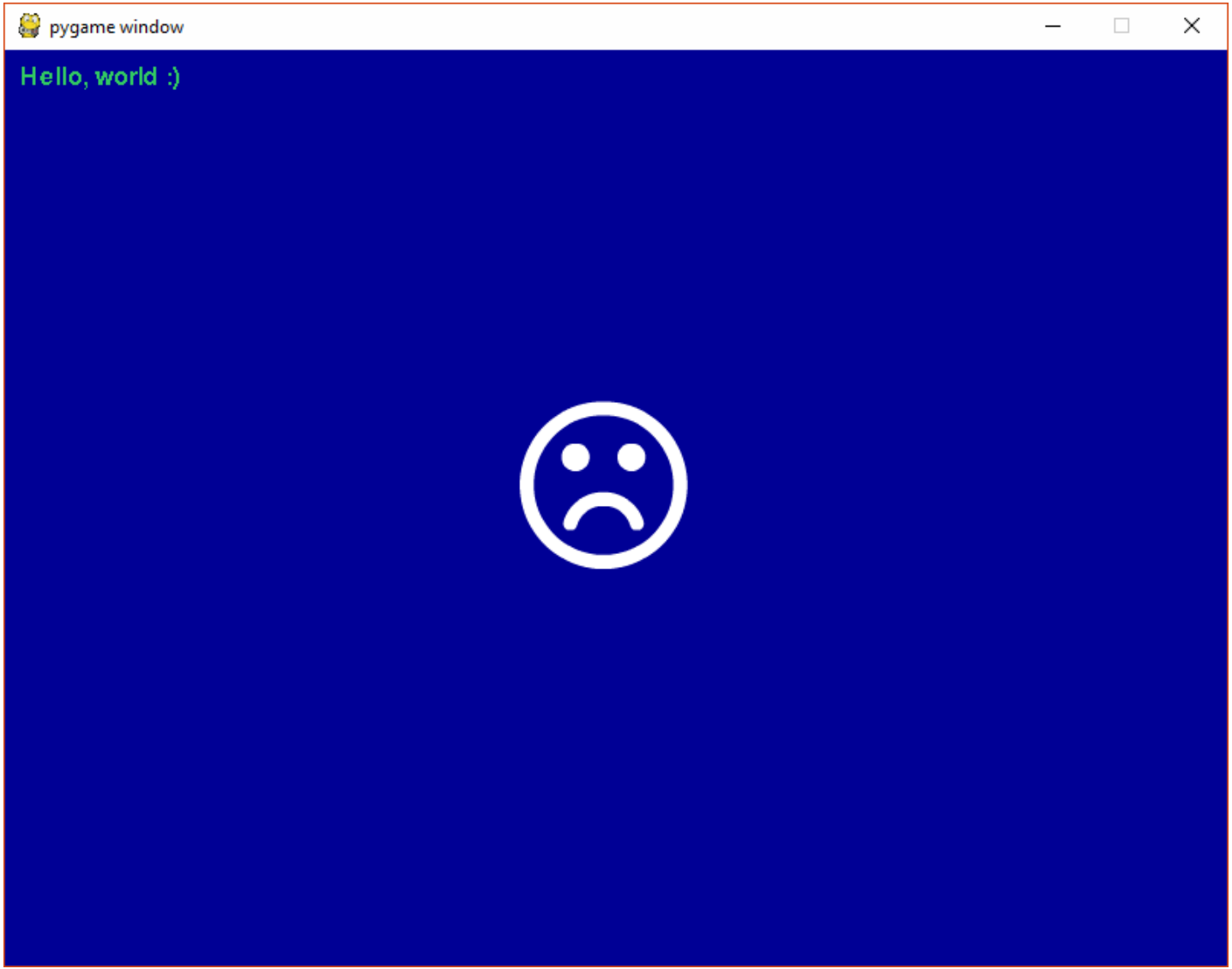
- main.py

```
...
screen.blit(text, (10, 10))

key = False
for event in pygame.event.get():
    if event.type == pygame.KEYDOWN:
        key = True
if key:
    text = font.render("How are you ?", False, (255, 255, 255))
    screen.blit(text, (10, 10))

pygame.display.update()
...
```

- key = False / key = True : boolean
- Hitting a key is an event. Pygame get all event with pygame.event.get()
- Events have type. Pressing a keyboard key is triggering the KEYDOWN event.


pygame window
Hello, world :)

Loop

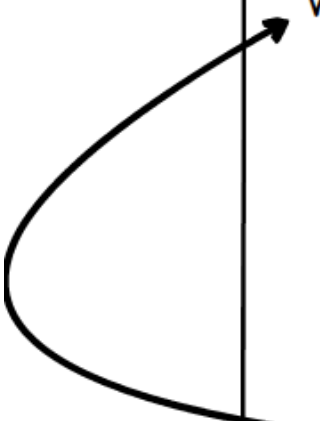
update

A loop is used for real time display

- main.js

```
...
screen.blit(text, (10, 10))

key = False
quit = False
while not(quit):
    for event in pygame.event.get():
        if event.type == pygame.KEYDOWN:
            key = True
    if key:
        text = font.render("How are you ?", False, (255, 255, 255))
        screen.blit(text, (10, 10))
        quit = True                                  # The loop will end
    pygame.display.update()
```



pygame window

How are you ?

Clean loop

- main.js

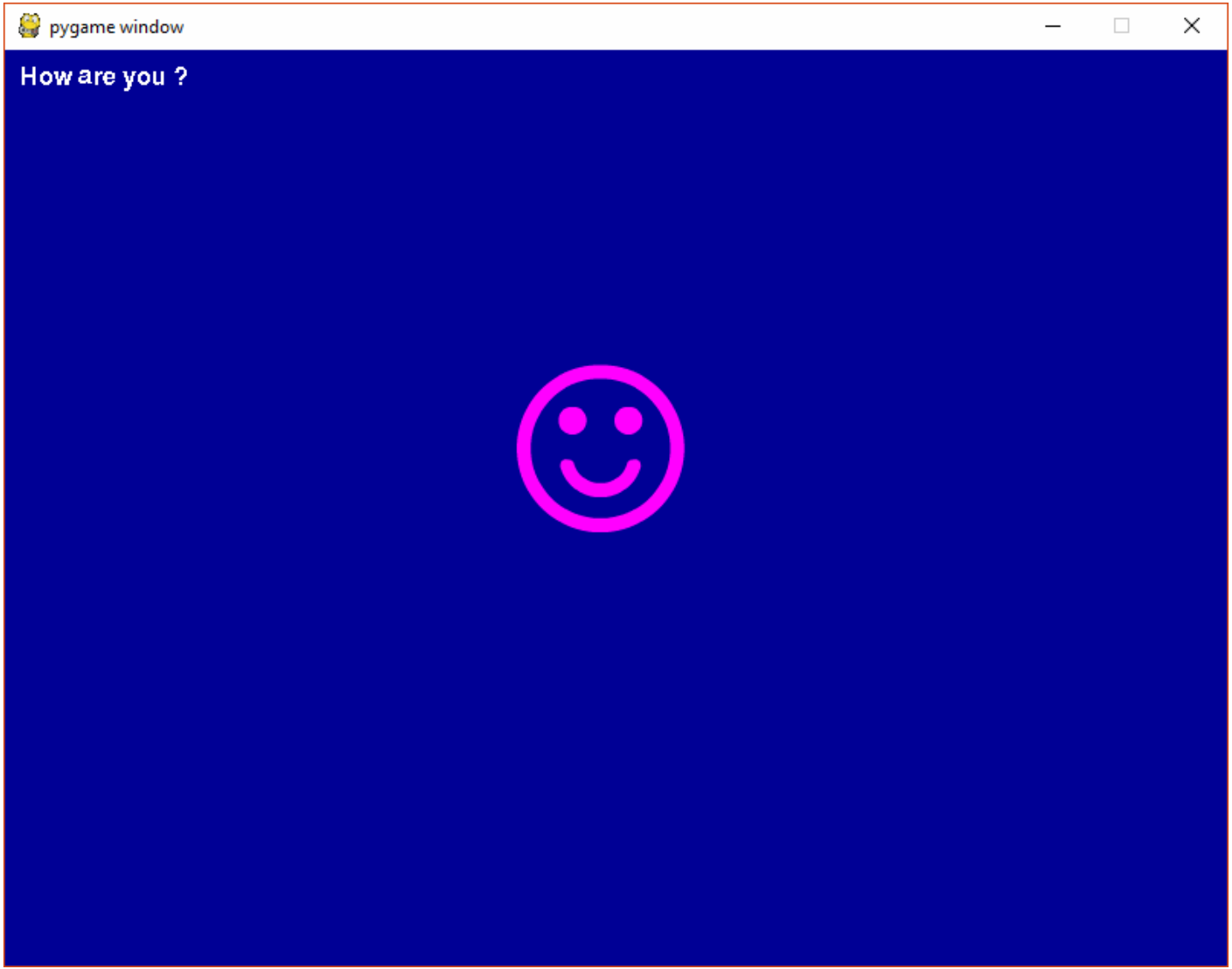
```
text = font.render("Hello, world :)", False, (50, 200, 100))
                                        # No more text blit

key = False
quit = False

while not(quit):
    for event in pygame.event.get():
        if event.type == pygame.KEYDOWN:
            key = True

    if key:
        text = font.render("How are you ?", False, (255, 255, 255))
        quit = True

    screen.fill((0, 0, 150))            # Erase screen
    screen.blit(text, (10, 10))         # Blit text contained in variable
    pygame.display.update()
```


pygame window
How are you ?

Quit event

Handle quit event

- main.py

```
import pygame, time, sys

...

for event in pygame.event.get():

        if event.type == pygame.QUIT:
            sys.exit()

        if event.type == pygame.KEYDOWN:
            key = True
```

A basic interaction

Final code

- main.py

```
import pygame, time, sys

def main():

    # Load
    pygame.init()
    screen = pygame.display.set_mode((800, 600))
    font = pygame.font.Font(None, 24)
    text = font.render("Hello, world :)", False, (50, 200, 100))
    key = False
    quit = False

    while not(quit):
        # Inputs
        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                sys.exit()
            if event.type == pygame.KEYDOWN:
                key = True

        # Update
        if key:
            text = font.render("How are you ?", False, (255, 255, 255))
            quit = True

        # Draw
        screen.fill((0, 0, 150))
        screen.blit(text, (10, 10))
        pygame.display.update()

    time.sleep(2) # End program after 2 seconds

if __name__ == "__main__":
    main()
```

Basics of a video game

```
window {
    load();

    loop {
        inputs();
        update();
        draw();
    }
}
```