



# ***Programación Avanzada***

## **IIC2233 2024-2**

Hernán Valdivieso - Daniela Concha - Francisca Ibarra - Lucas Van Sint Jan - Francisca Cattán



# Anuncios

1. Ya está publicada la T4 que pone en práctica este contenido.
  2.  Todo debe ser programado por ustedes, no con el apoyo de otras herramientas (IA, QTDesigner, etc) y/o persona .
  3. **¡No olviden hacer *push* de forma periódica!**
  4. Encuesta de Carga Académica.  
¡No se olviden!
-

# Interfaces gráficas II

# Concurrencia en PyQt

- QThread
- Thread

# Concurrencia en PyQt

## QThread

- Es un `Thread`, como el visto anteriormente, pero dentro de PyQt.
- Se utiliza igual que un `Thread`, es decir, definir un método **`run()`** y hacer **`.start()`** del `QThread`.
- También disponemos de **`QMutex`**, el análogo al `Lock` en PyQt. Este nos servirá para asegurar el acceso a un recurso crítico.

# Concurrencia en PyQt

## QThread vs Thread

- Ambos permiten aplicar concurrencia en PyQt.
- Thread no puede contener señales como atributo de clase y QThread si.
- Ambos pueden tener señales como atributos de instancia.
- Queda a tu criterio cuál de los 2 usar de aquí en adelante, puedes utilizar únicamente uno o intentar usar ambos. ¡Tú eliges 😁 !

# Concurrencia en PyQt

## QThread vs Thread

- Ambos permiten aplicar concurrencia en PyQt.
- Thread no puede contener señales como atributo de clase y QThread si.
- Ambos pueden tener señales como atributos de instancia.
- Queda a tu criterio cuál de los 2 usar de aquí en adelante, puedes utilizar únicamente uno o intentar usar ambos. ¡Tú eliges 😊 !
- ***Spoiler:*** aunque elijas usar únicamente uno, en el examen podemos preguntar por cualquiera de los 2 o ambos 🙊 .

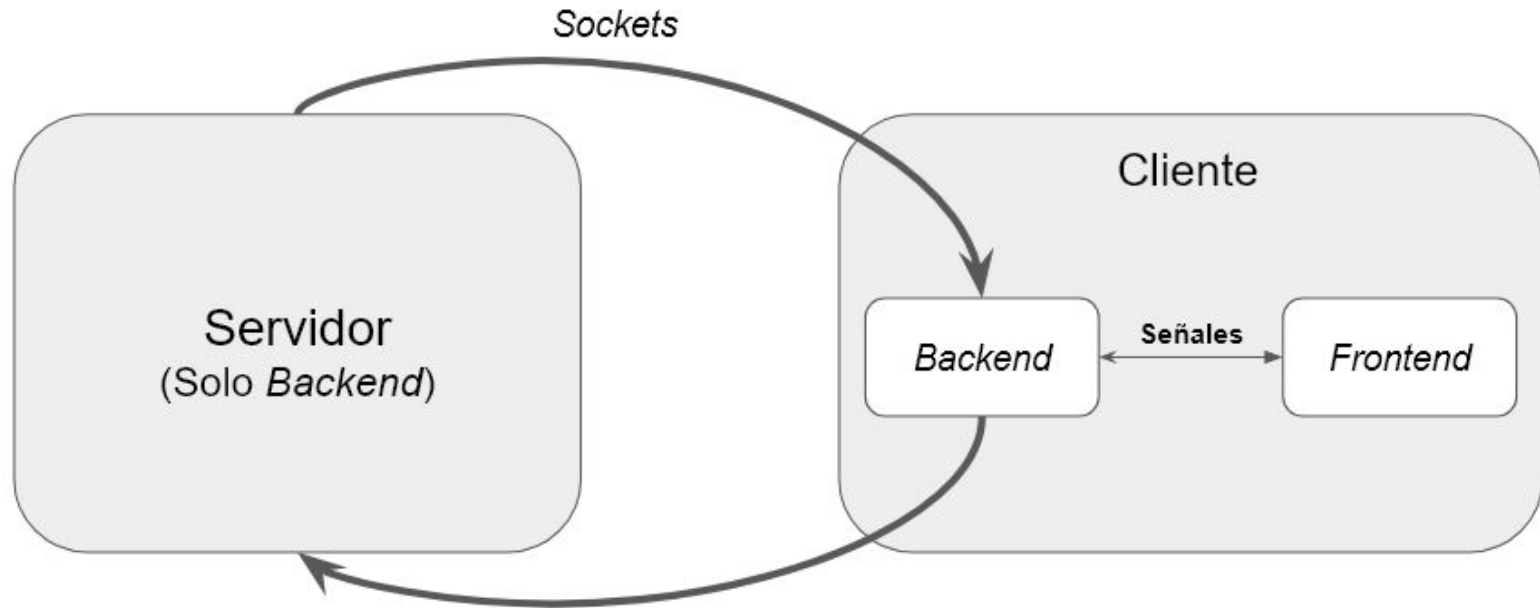
# Integración

- Networking e IG

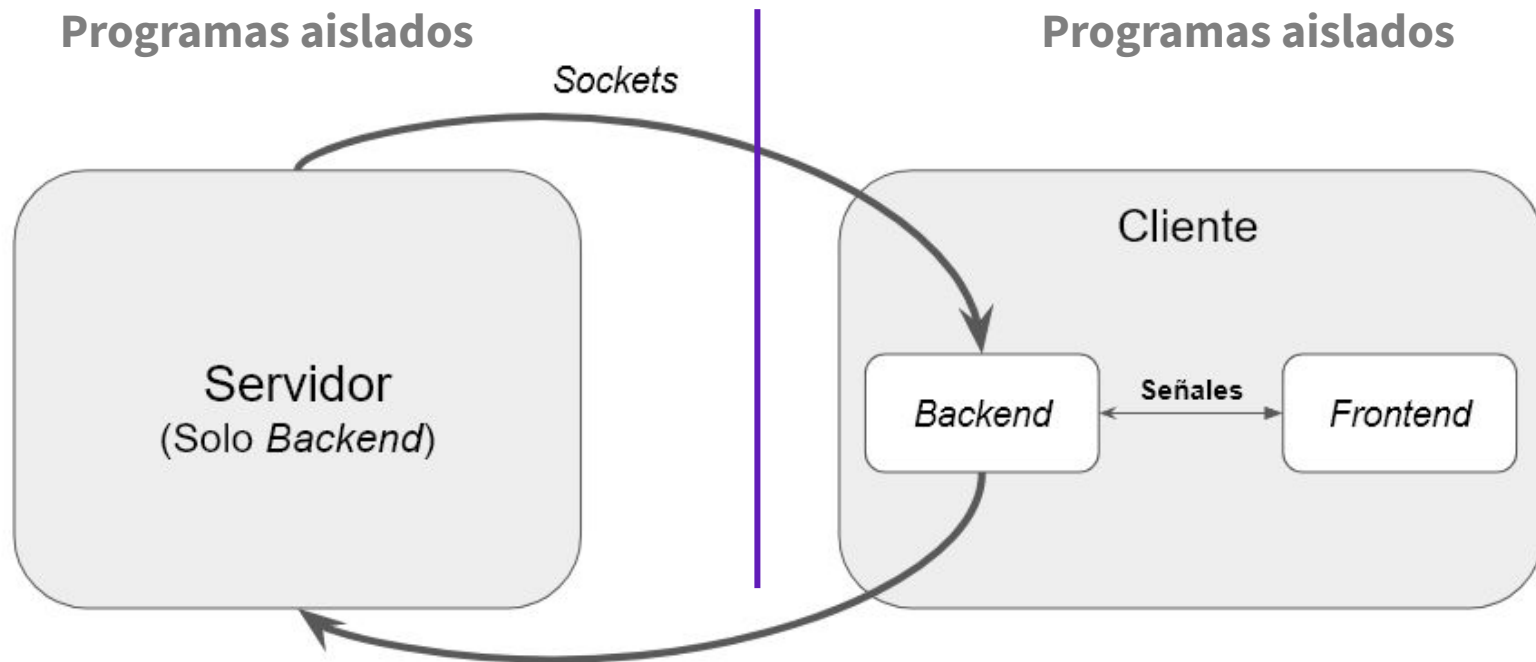
---



# Integración Networking y PyQt

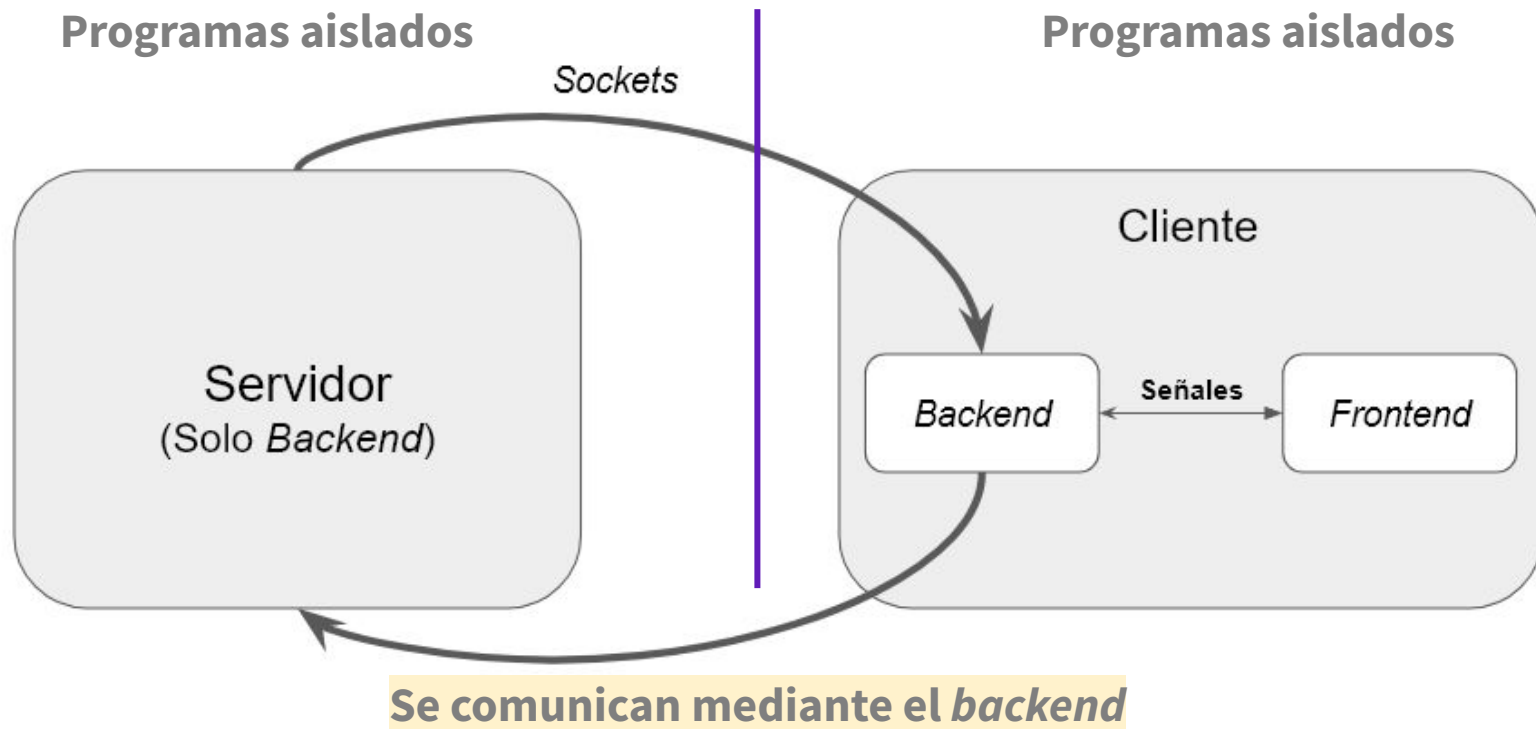


# Integración Networking y PyQt

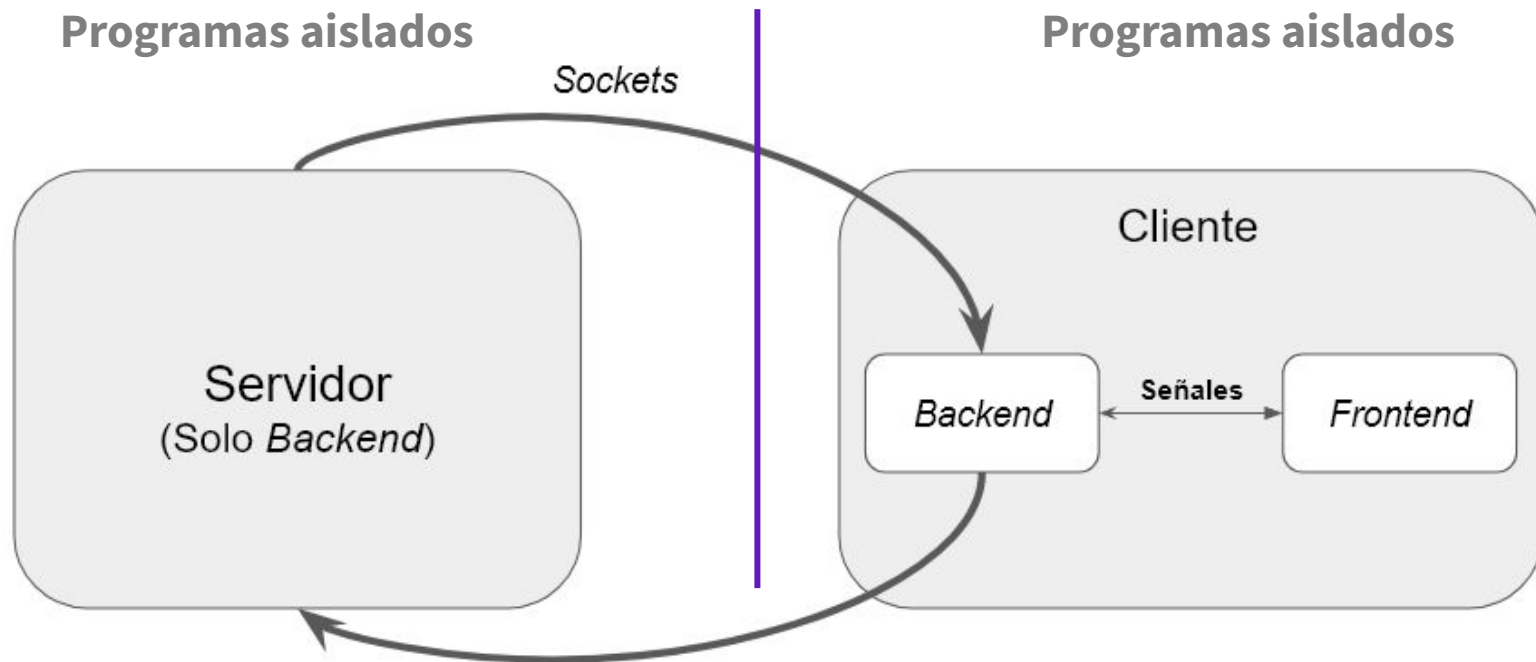


**Servidor y Cliente podrían estar en lenguajes diferentes**

# Integración Networking y PyQt



# Integración Networking y PyQt



**El servidor no debería tener dependencia con el *frontend***

# Misceláneos

- `isAutoRepeat`
- `QMediaPlayer`
- `QSoundEffect`

---

# Misceláneos

## isAutoRepeat

- Método de un evento en KeyPressEvent (QKeyEvent).
- Este método retorna un booleano que:
  - Toma el valor de **False** si el evento gatillado fue producto de presionar por **primera** vez una tecla.
  - Toma el valor de **True** si es un evento producto de **mantener** presionada la tecla.
- Útil cuando solo deseamos mandar una señal cuando se presiona la tecla, pero si el usuario la mantiene presionada, no ocurra nada.

# Misceláneos

## QSoundEffect

- Objeto de PyQt para reproducir archivos **.wav**.

```
self.media_player_wav = QSoundEffect(self) # Creo reproductor
file_url = QUrl.fromLocalFile(join("sounds", "musica.wav")) # Creo un path
media_player_wav.setSource(file_url) # Indico el path al reproductor
media_player_wav.play() # Reproducir
```

# Misceláneos

## QMediaPlayer

- Objeto de PyQt para reproducir archivos **.mp3**.
- Puede salir un *warning* en consola. No te preocupes 😊.

```
self.media_player = QMediaPlayer(self) # Creo reproductor
path = os.path.abspath(join("sounds", "musica.mp3")) # Creo un path
content = QMediaContent(QUrl.fromLocalFile(path)) # Creo un objeto contenido
self.media_player.setMedia(content) # Le doy el contenido al reproductor
self.media_player.play() # Reproducir
```



# Misceláneos

## Versiones de librerías

- Siempre tengan ojo de la versión de librerías e incluso de Python ocupada.
- Entre una versión y otra, pueden cambiar cosas que uno no se espera.

### PyQt5 (versión del curso)

```
self.player = QMediaPlayer(self)
content = QMediaContent(QUrl.fromLocalFile(os.path.abspath("sounds/musica.mp3")))
self.player.setMedia(content)
```

### PyQt6 (otra versión)

```
self.player = QMediaPlayer(self)
self.player.setAudioOutput(QAudioOutput(self))
self.player.setSource(QUrl.fromLocalFile("sounds/musica.mp3"))
```

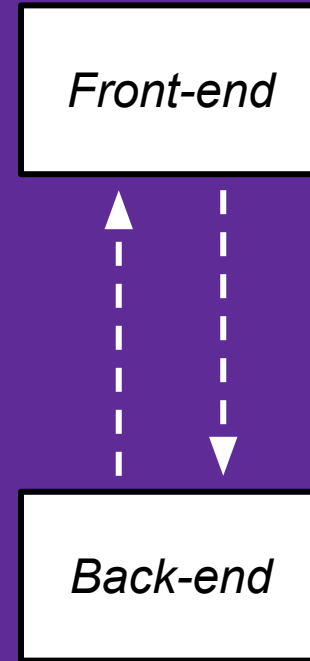
# Último mensaje

- Ahora que hay muchas clases en juego.

---

# REPETIMOS

Tengan mucho  
cuidado con las  
dependencias  
circulares



# Tengan mucho cuidado con las dependencias circulares

```
class Frontend:  
    def __init__(self):  
        self.backend = Backend(self)  
  
class Backend:  
    def __init__(self, frontend):  
        self.frontend = frontend
```



# Tengan mucho cuidado con las dependencias circulares

```
class Frontend:
```

```
    def __init__(self):
```

```
        self
```

```
class
```

**¡Usen señales!**

```
        self, frontend):
```

```
        self.frontend = frontend
```



# Pregunta de Evaluación Escrita

## Tema: Interfaz Gráfica (Midterm 2023-2)

13. ¿Cuál o cuáles de estas funcionalidades **pertenecen** al *front-end* de un programa?

- I. Almacenar los puntajes en un archivo.
- II. Recibir un nombre de usuario.
- III. Validar la contraseña del usuario.
- IV. Reproducir una canción.
- V. Cargar el logo del programa.

- A) I y III
- B) I, II y III
- C) II y V
- D) II, IV y V
- E) II, III y V

# Pregunta de Evaluación Escrita

## Tema: Interfaz Gráfica (Midterm 2023-2)

13. ¿Cuál o cuáles de estas funcionalidades **pertenecen** al *front-end* de un programa?

- I. Almacenar los puntajes en un archivo.
- II. Recibir un nombre de usuario.
- III. Validar la contraseña del usuario.
- IV. Reproducir una canción.
- V. Cargar el logo del programa.

- A) I y III
- B) I, II y III
- C) II y V
- D) II, IV y V**
- E) II, III y V

# ***Programación Avanzada***

## **IIC2233 2024-2**

Hernán Valdivieso - Daniela Concha - Francisca Ibarra - Lucas Van Sint Jan - Francisca Cattán

