PROGRAMACION FUNCIONAL

RODRIGUEZ MORENO MARCO ANTONIO

Aarón Hernández García

Actividad: 2.2 (Ejemplo de cada tipo de programación)

28/01/202

1. Programación imperativa

Definición:

Se basa en dar instrucciones claras y secuenciales sobre *cómo* realizar una tarea.

Ejemplo laboral:

En un proyecto de **landing page**, puedes usar JavaScript para manipular el DOM:

```
const button = document.getElementById('submit');
button.addEventListener('click', () => {
    const name = document.getElementById('name').value;
    alert(`Hola, ${name}`);
});
```

2. Programación declarativa

Definición:

Describe qué se debe hacer en lugar de cómo hacerlo.

Ejemplo laboral:

En diseño web, puedes usar HTML y CSS para declarar estilos sin especificar cómo renderizarlos:

```
<div class="button">Enviar</div>
<style>
  .button {
    background-color: #4CAF50;
    color: white;
    padding: 10px 20px;
    border-radius: 5px;
    text-align: center;
    cursor: pointer;
  }
</style>
```

3. Programación orientada a objetos (POO)

Definición:

Organiza el código en objetos que combinan datos y comportamiento.

Ejemplo laboral:

En tu proyecto de **MangaBox** con ASP.NET Core:

```
public class Manga {
    public string Title { get; set; }
    public string Author { get; set; }
    public int Chapters { get; set; }

    public void DisplayInfo() {
        Console.WriteLine($"{Title} by {Author}, {Chapters} chapters");
    }
}
```

4. Programación funcional

Definición:

Se enfoca en funciones puras y evita el estado compartido y efectos secundarios.

Ejemplo laboral:

En un proyecto de consulta de restaurantes con JavaScript:

```
const restaurants = [
    { name: "Trama Café", rating: 4.8 },
    { name: "Cine Bujazam", rating: 4.5 }
];

const highlyRated = restaurants.filter(r => r.rating > 4.6);
console.log(highlyRated);
```

5. Programación reactiva

Definición:

Se centra en flujos de datos asíncronos y el manejo de eventos.

Ejemplo laboral:

En tu aprendizaje de Flutter, usando un flujo reactivo con Stream:

```
import 'dart:async';

void main() {
  final StreamController<String> controller = StreamController();
  controller.stream.listen((data) => print('Evento recibido: $data'));
  controller.add('Nuevo mensaje desde Flutter');
}
```

Relación laboral

En tu experiencia, cada paradigma se aplica a diferentes contextos:

- **Imperativa:** Tareas específicas, como manipular el DOM.
- **Declarativa:** Desarrollo de interfaces con HTML y CSS o herramientas modernas como React.
- **POO:** Modelado de datos y estructuras en ASP.NET Core o Flutter.
- **Funcional:** Procesamiento de datos, como filtros y transformaciones.
- **Reactiva:** Manejo de eventos en tiempo real en aplicaciones móviles o webs interactivas.

Referencias

Sebesta, R. W. (2020). Concepts of Programming Languages (12th ed.). Pearson Education.

Un libro clásico para entender los fundamentos de los paradigmas de programación.

Staltz, A. (2019). The introduction to Reactive Programming you've been missing. Available online: ReactiveX.io

Un excelente recurso gratuito para comprender programación reactiva con RxJS.