



PROCESO DE GESTIÓN DE FORMACIÓN PROFESIONAL INTEGRAL

FORMATO GUÍA DE APRENDIZAJE

IDENTIFICACIÓN DE LA GUÍA DE APRENDIZAJE

- **Denominación del Programa de Formación:** TGO Análisis y Desarrollo de Sistemas de Información
- **Código del Programa de Formación:** 228106 V102
- **Nombre del Proyecto:** Construcción de un sistema de información que cumpla con los requerimientos del cliente en procesos que se lleven a cabo en el sector productivo del departamento de Caldas
- **Fase del Proyecto:** ANÁLISIS
- **Actividad de Proyecto:** Seleccionar la alternativa de solución que cumpla con los requerimientos establecidos por el cliente
- **Competencia:** Analizar los requisitos del cliente para construir el sistema de información.
- **Resultados de Aprendizaje Alcanzar:** Interpretar el informe de requerimientos, para determinar las necesidades tecnológicas en el manejo de la información, de acuerdo con las normas y protocolos establecidos en la empresa
- **Duración de la Guía:** 6 horas

2. PRESENTACIÓN

El Lenguaje Unificado de Modelado (UML) es un estándar que se ha adoptado a nivel internacional y que permite especificar, visualizar y documentar esquemas de sistemas de software orientado a objetos. Representa una serie de normas y estándares gráficos que ayuda a visualizar el diseño y a hacerlo más accesible para otros. UML está diseñado para su uso con software orientado a objetos, y tiene un uso limitado en otro tipo de cuestiones de programación.

En esta guía los aprendices encontrarán conceptos generales acerca de UML y su aplicación en el diseño de software, utilizando Dart como lenguaje de programación orientado a objetos. El aprendiz encontrará conceptos teóricos de programación orientada a objetos, estándares utilizados, elementos del diagrama de clases, casos de estudio y ejemplos.

Dart es un lenguaje de programación orientado a objetos (OOP, por sus siglas en inglés). Esto significa que se basa en conceptos como clases y objetos para estructurar el código. En esta guía el aprendiz identificará los aspectos clave de la orientación a objetos en el lenguaje de programación Dart, lo cual le permitirá tener bases de programación sólidas frente a este paradigma de programación.



3. FORMULACIÓN DE LAS ACTIVIDADES DE APRENDIZAJE

- **Descripción de la(s) Actividad(es)**
 - **Actividades de aprendizaje**
 - Instalación de Dart
 - Reconocimiento del entorno de programación con el lenguaje de programación Dart
 - Identificación de los aspectos más importantes del paradigma de Programación Orientada a Objetos
 - **Actividad de reflexión inicial**

Identificación de objetos

El grupo ADSO < XXX >, desea realizar una actividad de esparcimiento en el sector de Santagueda en VillaSena. Los aprendices deben tener en cuenta los productos e implementos que se deben llevar. Los productos se van a comprar en un almacén de cadena, por ejemplo, el “ARA” el “D1” o “EL AHORRO”, en el cual se venden accesorios para vehículos, ropa, electrodomésticos, alimentos y bebidas entre otros.

Para el paseo se necesitarán productos según su:

- | | |
|--------------------------|-------------------------|
| a) Categoría de Producto | d) Color |
| b) Subcategoría producto | e) Tamaño |
| c) Funcionalidad | f) Productos por Marcas |



- ¿Cuál podría ser la clasificación que se le puede dar dichos productos?
- ¿Qué productos se pueden clasificar en distintos grupos?
- ¿Qué productos tienen características semejantes y cuáles?

La anterior actividad debe realizarse en grupos de 3 aprendices durante la sesión de formación, una vez finalizada se socializará la actividad entre todos y el instructor aclarará las inquietudes presentadas por los aprendices.

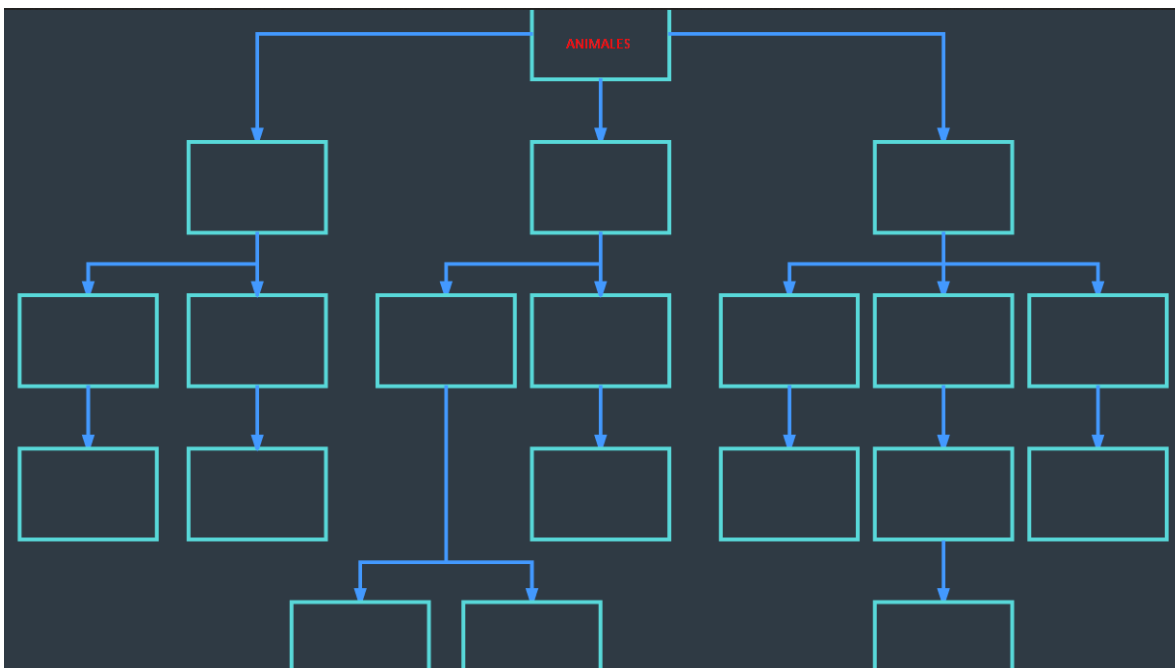
Actividades de contextualización e identificación de conocimientos necesarios para el aprendizaje.

Para esta actividad se deben tener en cuenta las siguientes dos imágenes y generar una especie de mapa conceptual que muestre los grupos y subgrupos encontrados, determinando las características que cada grupo posee



Animales	Vehículos de Transporte
	

A continuación, se muestra una base para ANIMALES, que se constituye como nodo principal, y se van desplegando las subcategorías. Lo mismo se haría con VEHÍCULOS DE TRANSPORTE





Actividad 01

Ahora los aprendices deberán realizar en clase la lectura del siguiente documento:

- [ProgOrientadaObjetos.pdf](#)

Una vez terminada la lectura del material de trabajo anexo, el instructor aclarará las inquietudes presentadas por los aprendices. Como resultado de esta actividad los aprendices deberán realizar un resumen con las ideas más importantes y conocerán los conceptos generales de la programación orientada a objetos.

Adicionalmente y para profundizar en los conceptos de POO, los aprendices deberán leer el libro: [programacion-orientada-a-objetos-luis-joyanes-aguilar.pdf](#). La lectura se realizará desde las páginas 16 – 28.

Con base en la lectura realizada los aprendices realizarán la siguiente actividad:

Actividad 02

Responder y socializar las siguientes preguntas:

- ¿Cuáles son las fases del ciclo de software?
- ¿Cuál es la fase que más costo genera en un proyecto de software?
- ¿Qué es la programación orientada a objetos?
- ¿Qué es abstracción?
- ¿Qué es encapsulación?
- ¿Qué es polimorfismo? Explique cómo se aplica el polimorfismo en la clase mamífero con la función Comer()
- ¿Qué es herencia?
- ¿Cuál es la diferencia entre Clase y Objeto?

Se realizará con los aprendices una socialización sobre las respuestas a las preguntas propuestas, en la cual el instructor resolverá inquietudes y explicará conceptos y principios básicos de la POO:

Actividades de apropiación del conocimiento (Conceptualización y Teorización).

En esta sección el aprendiz encontrará la documentación teórica acerca de cómo realizar un Diagrama de Clase teniendo en cuenta los conceptos, estándares y elementos mínimos que lo forman.

En la programación de computadores, se pueden encontrar diferentes lenguajes que se pueden utilizar para desarrollar aplicaciones o programas para computador. En algunas ocasiones, las diferencias entre los lenguajes de programación se pueden clasificar según las características que cada uno posee, un ejemplo se puede ver en cuanto a los estilos de baile que tienen las personas, por ejemplo, algunos bailarían salsa, otros vallenato, otros Electrónica y otros estilos o clases de bailes que existen en la actualidad.



En el área de la computación, existen diferentes estilos de programación llamados metodologías. “Una metodología de programación es un conjunto o sistema de métodos, principios y reglas que permiten enfrentar de manera sistemática el desarrollo de un programa que resuelve un problema algorítmico. Estas metodologías generalmente se estructuran como una secuencia de pasos que parten de la definición del problema y culminan con un programa que lo resuelve”, (Universidad Nacional de Colombia, s.f.). Algunas de las metodologías son la Estructurada, Orientada a Eventos, Orientada a Objetos.

La Programación Orientada a Objetos (POO) consiste en identificar, en el problema al cual se le quiere dar una solución computacional por medio de un software, una serie de elementos que tiene presente esta metodología entre los cuales se pueden listar el encapsulamiento, polimorfismo, sobrecarga y herencia. (Aguilar, 1996)

Cuando se le quiere dar solución a un problema de forma computacional, recuerde que se deben seguir una serie de pasos o etapas ordenadas para llegar a dicho objetivo. Las etapas pueden ser planeación, análisis, diseño, desarrollo e implantación de la solución. En esta guía vamos a enfocarnos en la etapa de Diseño tomando como base esa programación Orientada a Objetos específicamente el diagrama de clases, los conceptos necesarios, sus elementos gráficos y estándares.

En la programación orientada a objetos la base de todo es el **Objeto**. Un objeto es una entidad que se encuentra compuesta por una serie de características y funciones que desempeña dentro de un contexto, por ejemplo, en la actividad anterior, se puede tomar como un objeto la imagen del león. Para este objeto, el león puede tener las siguientes características: color, altura, edad, tipo de alimentación, etc..., las funciones se pueden referir a correr, dormir, comer, perseguir, acechar y otras.

Cuando en un problema se observa que hay objetos que pueden ser agrupados según su funcionalidad o sus características dentro de su posible solución, se dice que ese grupo o clasificación se llama **Clase**. Retomando la actividad anterior, se puede ver algunas clases presentes en el ejercicio como la clase Animal, Felino, Acuático, Terrestre, Doméstico y otras. Note que los nombres de clases antes mencionados todos tienen la primera letra en Mayúscula, esto se debe al estándar que se deben manejar en programación orientada a objetos, para que cuando un programador vea ese término, de inmediato lo asocia con la definición de una clase.

Dado el ejemplo anterior, se puede definir una clase como una plantilla que agrupa elementos que tienen las mismas características y funciones en un problema determinado.

Y con el ejemplo del objeto, este se puede definir como una instancia de la clase o la plantilla que se definió. Cuando se menciona instancia es asignarle datos a todos los atributos que tiene la clase, por es cuando utilizo diferentes datos para esos atributos, lo que se está haciendo es creando diferentes objetos de esa clase.

Ejemplos:

Clase Felino

Atributos (características): altura, peso, nombre, edad

Métodos (Funciones): correr, dormir, caminar, comer

Objeto: 30cms, 15kg, Michin, 2 años



```
1 class Persona {
2     String nombre;
3     int edad;
4
5     // Constructor
6     Persona(this.nombre, this.edad);
7
8     // Método
9     void mostrarInfo() {
10         print('Nombre: $nombre, Edad: $edad');
11     }
12 }
13
14 void main() {
15     var persona1 = Persona('Juan', 30);
16     persona1.mostrarInfo(); // Output: Nombre: Juan, Edad: 30
17 }
```

Clase en Java:

```
public class Coche {
    private String color;
    private int velocidad;
    private float tamaño;
    // Estado

    public Coche (String color, int velocidad, float tamaño){
        this.color = color;
        this.velocidad = velocidad;
        this.tamaño = tamaño;
    }
    // Constructor

    public void avanzar(){}
    public void parar(){}
    public void girarIzquierda(){}
    public void girarDerecha(){}
    // Comportamiento
}
```

Objeto:

```
public static void main (String[] args){
    Coche miCoche = new Coche ("verde", 80, 3.2f);
    Coche tuCoche = new Coche ("rojo", 120, 4.1f);
    Coche suCoche = new Coche ("amarillo", 100, 3.4f);
}
```

Implementación Clase en Dart



```
1 class Coche {
2     String color;
3     int velocidad;
4     double tamaño;
5
6     // Constructor
7     Coche(this.color, this.velocidad, this.tamaño);
8
9     // Método para avanzar
10    void avanzar() {
11        print('El coche de color $color está avanzando a $velocidad km/h.');
```

Objeto:

```
1 void main() {
2     // Crear una instancia de la clase Coche
3     Coche miCoche = Coche('Rojo', 60, 4.5);
4
5     // Llamar a los métodos de la instancia
6     miCoche.avanzar();
7     miCoche.girarIzquierda();
8     miCoche.parar();
9     miCoche.girarDerecha();
10 }
```

Setters y Getters

- **Getters:** Los métodos get permiten acceder a los valores de los atributos privados desde fuera de la clase.
- **Setters:** Los métodos set permiten modificar los valores de los atributos privados desde fuera de la clase, y pueden incluir lógica adicional para validar los nuevos valores.



Uso de los getters y setters:

En el método `main()`, se muestra cómo utilizar los métodos `get` y `set` para acceder y modificar los atributos del objeto `miCoche`.

Esta estructura proporciona un control más preciso sobre cómo se accede y se modifica el estado interno de los objetos, permitiendo incluir validaciones y lógica adicional cuando sea necesario.



```
1 class Coche {
2     String _color;
3     int _velocidad;
4     double _tamanno;
5
6     // Constructor
7     Coche(this._color, this._velocidad, this._tamanno);
8
9     // Getter para color
10    String get color {
11        return _color;
12    }
13
14    // Setter para color
15    set color(String nuevoColor) {
16        _color = nuevoColor;
17    }
18
19    // Getter para velocidad
20    int get velocidad {
21        return _velocidad;
22    }
23
24    // Setter para velocidad
25    set velocidad(int nuevaVelocidad) {
26        if (nuevaVelocidad >= 0) {
27            _velocidad = nuevaVelocidad;
28        } else {
29            print('La velocidad no puede ser negativa.');
```

```
30        }
31    }
32
33    // Getter para tamaño
34    double get tamanno {
35        return _tamanno;
36    }
37
38    // Setter para tamaño
39    set tamanno(double nuevoTamanno) {
40        if (nuevoTamanno > 0) {
41            _tamanno = nuevoTamanno;
42        } else {
43            print('El tamaño debe ser mayor que 0.');
```

```
44        }
45    }
46
47    // Método para avanzar
48    void avanzar() {
49        print('El coche de color $_color está avanzando a $_velocidad km/h.');
```

```
50    }
51
52    // Método para parar
53    void parar() {
54        print('El coche de color $_color ha parado.');
```

```
55        _velocidad = 0; // Asumimos que el coche se detiene completamente.
56    }
57
58    // Método para girar a la izquierda
59    void girarIzquierda() {
60        print('El coche de color $_color está girando a la izquierda.');
```

```
61    }
62
63    // Método para girar a la derecha
64    void girarDerecha() {
65        print('El coche de color $_color está girando a la derecha.');
```

```
66    }
67 }
68
69 void main() {
70     // Crear una instancia de la clase Coche
71     Coche miCoche = Coche('Rojo', 60, 4.5);
72
73     // Usar los métodos getter y setter
74     print('Color del coche: ${miCoche.color}');
```

```
75     miCoche.color = 'Azul';
76     print('Nuevo color del coche: ${miCoche.color}');
```

```
77
78     print('Velocidad del coche: ${miCoche.velocidad}');
```

```
79     miCoche.velocidad = 80;
80     print('Nueva velocidad del coche: ${miCoche.velocidad}');
```

```
81
82     print('Tamaño del coche: ${miCoche.tamanno}');
```

```
83     miCoche.tamanno = 5.0;
```



- **Ambiente Requerido**

Ambiente de SISTEMAS con conexión eléctrica e internet

- **Materiales**

- Computadores (30)
- Sillas (3)
- Televisor (1)
- Resma tamaño carta (1)
- Marcadores (3)
- Lápiz (1)
- Lapicero (1)

4. ACTIVIDADES DE EVALUACIÓN

Evidencias de Aprendizaje	Criterios de Evaluación	Técnicas e Instrumentos de Evaluación
Evidencias de Conocimiento: Evidencias de Desempeño: Asistencia y participación activa en las diferentes actividades propuestas Evidencias de Producto: Respuestas y procedimiento de los talleres realizados	Crea la base de datos en el motor de base de datos seleccionado, siguiendo especificaciones técnicas del informe, según normas y protocolos de la empresa.	Observación: EXC_D_01 Valoración del Producto: EXC_P_01

5. GLOSARIO DE TÉRMINOS

Sistema de información: es un conjunto de elementos orientados al tratamiento y administración de datos e información, organizados y listos para su uso posterior, generados para cubrir una necesidad o un objetivo

Sistema operativo – Es un conjunto de programas que sirven para manejar un ordenador.

Software - El conjunto de programas, procedimientos y documentación asociado a un sistema informático.

Javascript: es un lenguaje de programación del lado del cliente que se utiliza con frecuencia en diseño WEB para generar efectos más complejos que no se puedan alcanzar usando HTML.



HTML: Siglas de las palabras inglesas: Hypertext Markup Language. Es decir, lenguaje de marcado de hipertexto. Lenguaje informático para crear páginas web. Conjunto de etiquetas o instrucciones que permiten estructurar el contenido de una web e incluir los hipervínculos o enlaces a otras páginas. Este lenguaje lo inventó en 1991 el Doctor Berners-Lee del CERN en Suiza.

HTTPS: Siglas de las palabras inglesas: HyperText Transfer Protocol Secure o versión segura del protocolo HTTP. Es el protocolo empleado para la transferencia de ficheros HTML cifrados que puedan contener información confidencial.

HTTP: siglas de las palabras inglesas: Hypertext Transfer Protocol. A saber en español: Protocolo de Transmisión de Hipertexto. Protocolo estándar de transferencia de hipertexto. Es decir: el protocolo de comunicaciones en el que está basado la Word Wide Web.

Script: es un archivo de órdenes o archivo de procesamiento por lotes. Es un programa usualmente simple, que por lo regular se almacena en un archivo de texto plano.

MySQL: es un sistema de gestión de bases de datos de código abierto que, junto con PHP, permite darle a las páginas web cierto dinamismo, es decir, disponer de manera adecuada los datos solicitados por los navegadores. Es un sistema multiplataforma y su uso está tan extendido en las bases de datos que podría considerarse un estándar.

SEO (Search Engine Optimisation) Optimización en buscadores: técnica utilizada para asegurar que una página Web es compatible con los motores de búsqueda y así tener la posibilidad de aparecer en las posiciones más altas en los resultados de búsqueda.

Diseño web adaptable (responsive web design): se llama así al diseño web de aquellas páginas que se adaptan al tamaño de la pantalla o ventana en que se despliegan, por medio del uso de, idealmente, un solo documento HTML y un solo documento CSS. Esto permite hacer una sola página web para smartphones, phablets, tablets y PC.

Diagrama o Modelo Entidad Relación (DER): denominado por sus siglas en inglés, E-R "Entity relationship", o del español DER "Diagrama de Entidad Relación") es una herramienta para el modelado de datos que permite representar las entidades relevantes de un sistema de información así como sus interrelaciones y propiedades

Bases de Datos (BD): es un banco de información que contienen datos relativos a diversas temáticas y categorizados de distinta manera, pero que comparten entre sí algún tipo de vínculo o relación que busca ordenarlos y clasificarlos en conjunto.

6. REFERENTES BIBLIOGRÁFICOS

- Documentos técnicos relacionados en la plataforma
- <https://dart.dev>
- <https://dartpad.dev/>
- <https://flutter.dev/>
- <https://docs.flutter.dev/get-started/install>
- <https://docs.flutter.dev/get-started/install/windows/web?tab=download>
- <https://code.visualstudio.com/docs/?dv=win64user>
- <https://docs.flutter.dev/ui/widgets/basics>
- <https://api.flutter.dev/flutter/material/Scaffold-class.html>



- <https://jsonplaceholder.typicode.com/>
- <https://gemini.google.com/?hl=es>
- <https://www.titular.com/blog/que-es-gemini-la-nueva-ia-de-google>
- <https://developer.android.com/studio/index.html>

7. CONTROL DEL DOCUMENTO

	Nombre	Cargo	Dependencia	Fecha
Autor (es)	Julian Salazar Pineda	Instructor	Centro de Procesos Industriales y Construcción	10 de Abril de 2024

8. CONTROL DE CAMBIOS (diligenciar únicamente si realiza ajustes a la guía)

	Nombre	Cargo	Dependencia	Fecha	Razón del Cambio
Autor (es)					