APUNTES

Instructor. CRISTIAN DARIO PIMIENTA: [cdpimientar@sena.edu.co](mailto:cdpimientar@sena.edu.co)

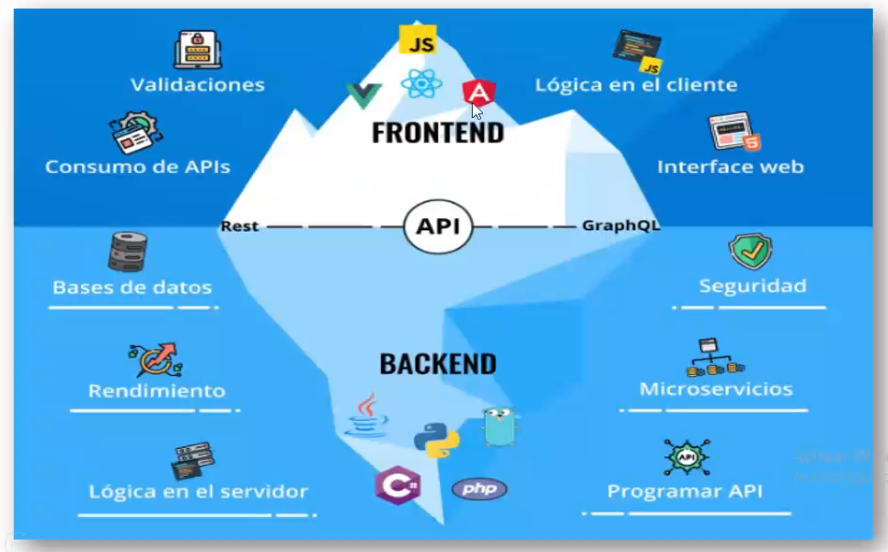
Instructor. ROGER HERNÁNDEZ: [rshernandez@sena.edu.co](mailto:rshernandez@sena.edu.co)

**SESIÓN UNO:**

1. Presentación de participantes
2. Ruta:



1. APP MONOLÍTIVA VS MICROSERVICIOS



1. Principios fundamentales de Dart

<https://dartpad.dev/>

|  |
| --- |
| TALLER 1:   1. **Suma de dos números**: Escribe un programa que pida al usuario dos números enteros y luego muestre la suma de esos dos números. 2. **Factorial de un número**: Desarrolla un algoritmo que calcule el factorial de un número entero ingresado por el usuario. El factorial de un número n se denota como n! y se calcula como el producto de todos los enteros positivos desde 1 hasta n. 3. **Ordenar una lista**: Crea un programa que ordene una lista de números enteros en orden ascendente. Puedes implementar algún algoritmo de ordenamiento como el método de la burbuja o de selección. 4. **Buscar un elemento en una lista**: Escribe un algoritmo que busque un número dado en una lista de números enteros. Si el número está presente en la lista, el programa debe indicar su posición; de lo contrario, debe mostrar un mensaje indicando que el número no se encontró. 5. **Contar palabras en una cadena de texto**: Desarrolla un algoritmo que cuente el número de palabras en una cadena de texto ingresada por el usuario. Una palabra se define como una secuencia de caracteres separados por espacios en blanco. 6. **Calcular el promedio de una lista de números**: Escribe un programa que calcule el promedio de una lista de números enteros ingresados por el usuario. 7. **Convertir temperatura**: Crea un algoritmo que convierta una temperatura dada en grados Celsius a grados Fahrenheit. La fórmula de conversión es: Fahrenheit = (Celsius × 9/5) + 32. 8. **Encontrar el número más grande en una lista**: Desarrolla un programa que encuentre el número más grande en una lista de números enteros ingresados por el usuario. 9. **Fibonacci**: Escribe un algoritmo que genere los primeros n números de la secuencia de Fibonacci, donde cada número es la suma de los dos números anteriores. Los primeros dos números de la secuencia son 0 y 1. 10. **Verificar si un número es primo**: Crea un programa que verifique si un número dado por el usuario es primo o no. Un número primo es aquel que es divisible únicamente por 1 y por sí mismo. |

**SESIÓN DOS:**

1. Recordando sesión uno:
   1. Arquitectura monolítica y microservicios (Front-End y Back-End)
   2. Dart: Listas, maps, colecciones y funciones.
2. Descargar, descomprimir e instalar para Windows / Web ‘[Flutter.dev](https://docs.flutter.dev/get-started/install/windows/web?tab=download)’ en el disco local C:. Dependiendo lo que escoja, descargará un instructivo o guía.
3. Editar variable de entorno en el PATH / Nuevo / ‘C:\flutter\bin’ / Aceptar
4. En el PowerShell, escribir ‘flutter’ - ‘flutter doctor’. Nota: Si se tiene instalador ‘Dart’, es mejor desinstalarlo, porque ‘Flutter’ ya tiene ‘Dart’
5. Instalar VSCode con las extensiones: Flutter - Flutter Widget Snippets - Awesome Widget Snippets - Flutter Color - Material Icon Theme
6. Crear proyecto: view / command palette (Ctrl + shift + p) / Flutter: New Project / Application
7. Iniciar el servidor (terminal):
   1. Run / Star debbuging
   2. “Parte superior derecha, Icono:” / “Run”
   3. flutter run **(terminal)**.
   4. ctrl + c (Cerrar el servidor) **(terminal)**.
8. Actividad Sesión\_2:
   1. Proyecto nuevo
   2. Ejecutar Chrome
   3. Visualización modo móvil
   4. Cambiar el color morado
   5. Cambiar la información del toolbar por tu nombre

**SESIÓN TRES:**

1. Recordando sesión dos:
   1. Instalación de Flutter, mi primer proyecto
2. Estructura de carpetas y archivos
   1. Todo lo que se ve en Flutter se llama ‘widget’ (Fragmentos de código que intervienen directamente con las interfaces gráficas). Existen dos tipos: 1] StatelessWidget, no tienen estado, no cambian en tiempo de ejecución (Ejp, etiquetas, textos); 2] StatefulWidget, si tienen estado (Ejp, botón).
   2. La aplicación arranca desde ‘lib/main.dart’ en la función ‘void main()’.
   3. Agregar función ‘main’ / Dentro ‘runApp’.
   4. Agregar estructuras de widgets con ‘sta’ (Ejemplo, StatelessWidget).
   5. En lugar del ‘Container’, reemplazar con MaterialApp’. Agregar los widget para ‘pintar’ la página inicial (Ejemplo, title y home: Text()).
   6. Reemplazar en el home: ‘Text()’ con ‘Column()’.
   7. Crear un estándar de organización a través del widget ‘Scaffold’.
   8. Nota: Se aconseja tener servidor de archivos (imágenes), db, aplicaciones, etc.
3. Trabajando con widgets:
   1. Column,
   2. Text,
   3. Scafold,
   4. MaterialApp,
   5. Appbar,
   6. Listtile,
   7. Icon,
   8. Image,
   9. elevateButton

**SESIÓN CUATRO:**

1. Recordando sesión tres:
   1. Trabajando con el archivo ‘lib/main’
2. Trabajando con widget ‘StatefulWidget’ y:
   1. Listview,
   2. Gridview,
   3. Card,
   4. Circleavatar,
   5. showmodalbuttonshep

**SESIÓN CINCO:**

1. Recordando sesión cuatro:
   1. Trabajando con el archivo ‘lib/main’
2. Trabajando con el widget ‘StatefulWidget’ y:
   1. Formularios
3. Se quitan las líneas azules agregando const a ‘Principal’ y ‘MaterialApp’ y otros.

**SESIÓN SEIS:**

1. Recordando sesión cinco:
   1. Trabajando con el archivo ‘lib/main’
2. Trabajando con widgets de:
3. Ruta Backend
4. Consumo de apis (página principal de Flutter):
   1. Convertir objeto flutter a Formato Json
   2. Crear carpeta y archivo ‘lib /api / ConsultarComentarios.dart’
   3. En la terminal, escribir ‘flutter pub add http’, para arreglar el problema de la dependencia para escribir ‘urls’
   4. A través de: <https://jsonplaceholder.typicode.com/>
5. Crear una api con php y llamarla desde la app de Flutter
6. Crear archivos y métodos para separar y organizar el código en main
7. Para exportar el proyecto, desde la terminal: Ejemplo, ‘flutter build apk’ para Android o ‘flutter build web –web-renderer html’
8. Y para subir a la app store (Ver documentación Flutter):