**Barranquilla 31 de julio de 2023**



**SERVICIO NACIONAL DE APRENDIZAJE “SENA”**

**CENTRO INDUSTRIAL Y DE AVIACION**

**Ficha Técnica:**

**Aprendiz:**

**Tutor:**

**Ejercicio:**

**2712261 – PROGRAMACION DE APLICACIONES PARA DISPOSITIVOS MOVILES**

**MANUEL ANDRES CARO VERGARA**

**CESAR AUGUSTO YEPES FLORES**

**[Conceptos bases de Dart](http://sena.territorio.la/tarea_tt.php?tarea=438997982" \o "Link de Evidencia)**

**¿Qué es Dart?**

Dart es un lenguaje de cliente optimizado para el desarrollo rápido de aplicaciones en cualquier plataforma. Su objetivo es demostrar el mas productivo lenguaje de programación para el desarrollo multiplataforma, junto con un flexible tiempo de ejecución de plataforma para los Frameworks de aplicaciones.

Dart está diseñado para un entorno técnico que es particularmente adecuado para el desarrollo de clientes, priorizando tanto el desarrollo (recarga en caliente con estado sub-segundo) como las experiencias de producción de alta calidad en una amplia variedad de objetivos de compilación (web, móvil y escritorio).

Dart también forma la base de Flutter . Dart proporciona el lenguaje y los tiempos de ejecución que impulsan las aplicaciones de Flutter, pero Dart también es compatible con muchas tareas básicas del desarrollador, como formatear, analizar y probar el código.

El lenguaje Dart es de tipo seguro; utiliza la verificación de tipo estático para garantizar que el valor de una variable siempre coincida con el tipo estático de la variable. A veces, esto se conoce como escritura de sonido. Aunque los tipos son obligatorios, las anotaciones de tipos son opcionales debido a la inferencia de tipos.

**Fundamentos de programación con Dart**

**Variables**

Las variables en dar pueden ser definidas con un tipo de dato implícitas, como el lenguaje es de tipo seguro, es decir, necesita que tipo de variables debe ser, cadena, booleano, numero, etc, y hay dos formas para declarar una variable de forma segura y es de forma explícita o implícita, la forma explicita es colocar el tipo de dato antes del nombre de la variable y luego inicializar, la forma implícita es declarar la variable e inicializarla inmediatamente con un valor primitivo y con el uso de las palabras clave.

// Variables de forma implicita

var name = 'Manuel';

const lastName = 'Caro';

// Variables de forma explicita

String hello = 'Hello World';

int age = 20;

**Operadores**

| **Description** | **Operator** | **Associativity** |
| --- | --- | --- |
| unary postfix | *expr*++    *expr*--    ()    []    ?[]    .    ?.    ! | None |
| unary prefix | -*expr*    !*expr*    ~*expr*    ++*expr*    --*expr*      await *expr* | None |
| multiplicative | \*    /    %  ~/ | Left |
| additive | +    - | Left |
| shift | <<    >>    >>> | Left |
| bitwise AND | & | Left |
| bitwise XOR | ^ | Left |
| bitwise OR | | | Left |
| relational and type test | >=    >    <=    <    as    is    is! | None |
| equality | ==    != | None |
| logical AND | && | Left |
| logical OR | || | Left |
| if null | ?? | Left |
| conditional | *expr1* ? *expr2* : *expr3* | Right |
| cascade | ..    ?.. | Left |
| assignment | =    \*=    /=   +=   -=   &=   ^=   *etc.* | Right |

| **Operador** | **Nombre** | **Significado** |
| --- | --- | --- |
| () | Aplicación de función | Representa una llamada de función. |
| [] | acceso de subíndice | Representa una llamada al []operador reemplazable; ejemplo: fooList[1]pasa el int 1a fooListpara acceder al elemento en el índice1 |
| ?[] | Acceso de subíndice condicional | Me gusta [], pero el operando más a la izquierda puede ser nulo; ejemplo: fooList?[1]pasa el int 1a fooListpara acceder al elemento en el índice 1a menos que fooListsea nulo (en cuyo caso la expresión se evalúa como nulo) |
| . | Acceso para miembros | Hace referencia a una propiedad de una expresión |
| ?. | Acceso de miembro condicional | Me gusta ., pero el operando más a la izquierda puede ser nulo; ejemplo: foo?.barselecciona la propiedad barde la expresión fooa menos fooque sea nulo (en cuyo caso el valor de foo?.bares nulo) |
| ! | Operador de aserción nula | Convierte una expresión en su tipo subyacente que no admite valores NULL y genera una excepción en tiempo de ejecución si la conversión falla; ejemplo: foo!.barafirma fooque no es nulo y selecciona la propiedad bar, a menos que foosea nulo, en cuyo caso se lanza una excepción de tiempo de ejecución |

**Estructuras de control**

**If/Else**

  int edad = 20;

  if (edad >= 18) {

    print('Es mayor de edad');

  } else {

    print('Es menor de edad');

  }

**If/Else/ Else If**

var estadoUsuario = 'activo';

  if (estadoUsuario == 'activo') {

    print('El usuario se encuentra en linea');

  } else if (estadoUsuario == 'espera') {

    print('El usuario esta en fila');

  } else {

    print('El usuario no se encuentra');

  }

**Switch**

int mes = 3;

  switch (mes) {

    case 1:

      print('El mes es Enero');

      break;

    case 2:

      print('El mes es Febrero');

      break;

    case 3:

      print('El mes es Marzo');

      break;

    case 4:

      print('El mes es Abril');

      break;

    // ...

    default:

      print('No se ha mencionado un numero de mes');

      break;

  }

**Ciclo For**

for (int numero in [1, 2, 3, 4, 5]) {

  print(numero);

}

for (int numero = 0; numero >= 5; numero++) {

  print(numero);

}

**Ciclo While**

var whileActive = 'active';

while (whileActive == 'active') {

  print('Un bucle infinito');

}