Patrón arquitectónico OO con Dart

Nombre y apellidos:	José Teodosio Lorente Vallecillos
Rol (director de proyecto/arquitecto/experto/programador):	Director de proyecto

Objetivos

- Aprender las características básicas de un nuevo lenguaje de programación OO por comparación con los más parecidos entre otros ya conocidos.

NOTA: Antes de empezar el taller debes completar los pasos previos, haciendo uso de las páginas web que se refieren y de cualesquiera otras que puedan ayudar a resolver cualquier problema de instalación y/o creación de un proyecto Dart.

PASO PREVIO: Instalación

Instalación del SDK Dart.- Podemos instalar solo Dart o instalar también Flutter y Android Studio, que necesitaremos para un taller posterior. Recomendamos la opción de instalar también Flutter y Android Studio.

- **OPCIÓN 1 (recomendada):**
  - **Instalación de Android Studio, Dart y Flutter**.-Instala el sdk de Flutter (https://docs.flutter.dev/get-started/install) y Android Studio y los plugins para Dart y Flutter (https://flutter.dev/docs/get-started/editor#androidstudio)
  - **Creación de proyecto Dart**.- Abre Android Studio y crea un proyecto Dart (File→New→New Flutter Project; selecciona Dart en el menú vertical de la izquierda). Se creará un fichero main.dart que no vamos a usar porque por ahora no estamos interesados en Flutter, sino en usar el IDE de Android Studio con Dart y Flutter para desarrollar en Dart con la salida por la consola. Borra su contenido.
  - **Creación de la clase principal**.- Crea la clase principal (File→New→ Dart file). Escribe en ese fichero un método main para imprimir “Hola mundo”.
  - **Ejecución de un proyecto.**- Ejecuta main.dart (Menú contextual: Run main.dart), y mira la salida en el panel de ejecución (Run) de Android Studio.
  - **Estructura del proyecto**.- Los ficheros dart se crean directamente en el raíz del proyecto. Vamos a crear un directorio lib (File→New→Directory) y el resto de los ficheros que crearemos para este taller los pondremos ahí.

- **OPCIÓN 2:**
  - **Instalación de Dart.-** También puedes optar por instalar solo Dart, siguiendo por ejemplo las indicaciones en https://dart.dev/get-dart o usar clados.ugr.es, donde ya está instalado.
  - **Uso de la interfaz de texto dart**.- Asegúrate que desde la consola puedes usar Dart (ejecuta $ *dart*). Si no es posible, deberás añadir al path el camino a dart. En el caso de haberlo instalado con Flutter, suele estar en [*flutter]*/common/flutter/bin/cache/dart-sdk/bin, siendo [*flutter]* el camino donde está instalado Flutter.
  - **Creación proyecto Dart.**- Puedes crear el típico proyecto HolaMundo, llamándolo *dart_taller*, haciendo:
    - $dart create -t console-full dart_taller
  - **Ejecución del proyecto**.- Sitúate en el raíz del proyecto y escribe:
    - $ dart run
  - **Estructura del proyecto**.- El fichero con el método main está situado en la carpeta bin. El resto de ficheros se sitúan en la carpeta lib.
  - **Cambios en el texto**.- Introduce algún cambio en el código y vuelve a ejecutar. A partir de aquí, puedes comenzar a trabajar en la implementación del problema arriba especificado.

Material disponible

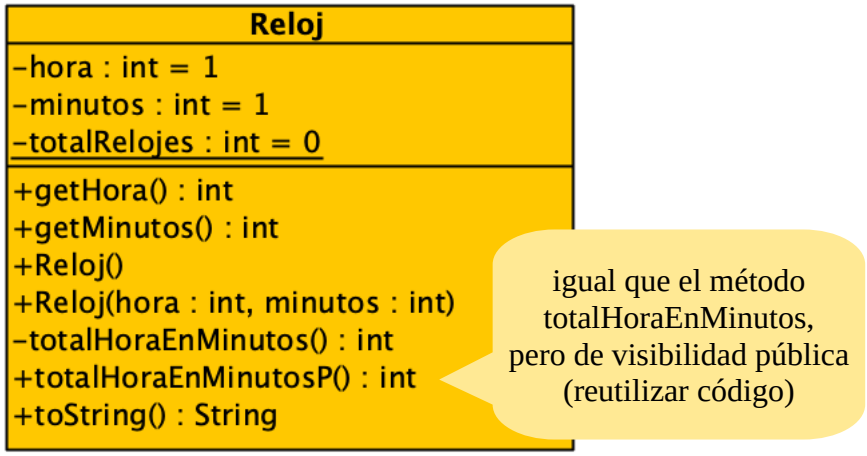
En la Entornos de trabajo y lenguajes→Dart de PRADO hay un esquema que puede ser de gran ayuda para comprender Dart a partir de Java y Ruby.

Descripción del problema

- Se debe implementar en Dart la clase Reloj (en lib/Reloj.dart) y las funciones de ámbito global *totalHorasCompletas* y *totalHoras* en el fichero (main.dart), según se describe a continuación:

  - *totalHorasCompletas*: A partir de la hora total en minutos dada como argumento, calcula las horas completas (división entera) y devuelve un entero

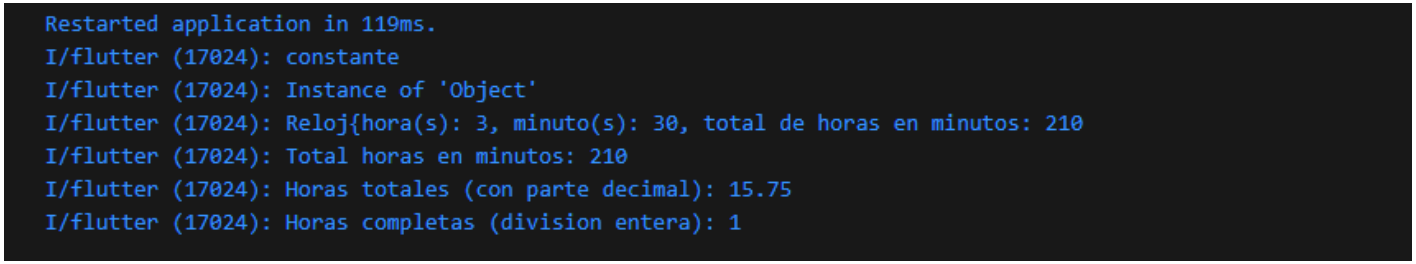
  - *totalHoras*: A partir de la hora total en minutos dada como argumento, calcula las horas totales con parte decimal. Devuelve un object genérico



- Escribe un programa principal (en bin/main.dart) con:
  - Una constante con valor “constante” y una variable genérica (un object) con valor “generica”
  - Un método main que:
    - imprima la constante y la variable
    - cree tres relojes (al menos uno debe crearse con el constructor básico)
    - imprima la hora completa (su estado, usando el método *toString*) y el total de horas en minutos
    - llame a los métodos globales *totalHoras* y *totalHorasCompletas* con la hora en minutos de alguno de los relojes e imprima los resultados

Resultados:

Inserta aquí una captura de pantalla con la salida de ejecutar el programa:



Se subirá este fichero a Prado con el cuadro anterior relleno.