ACCESO Y PROCESAMIENTO DE DATOS

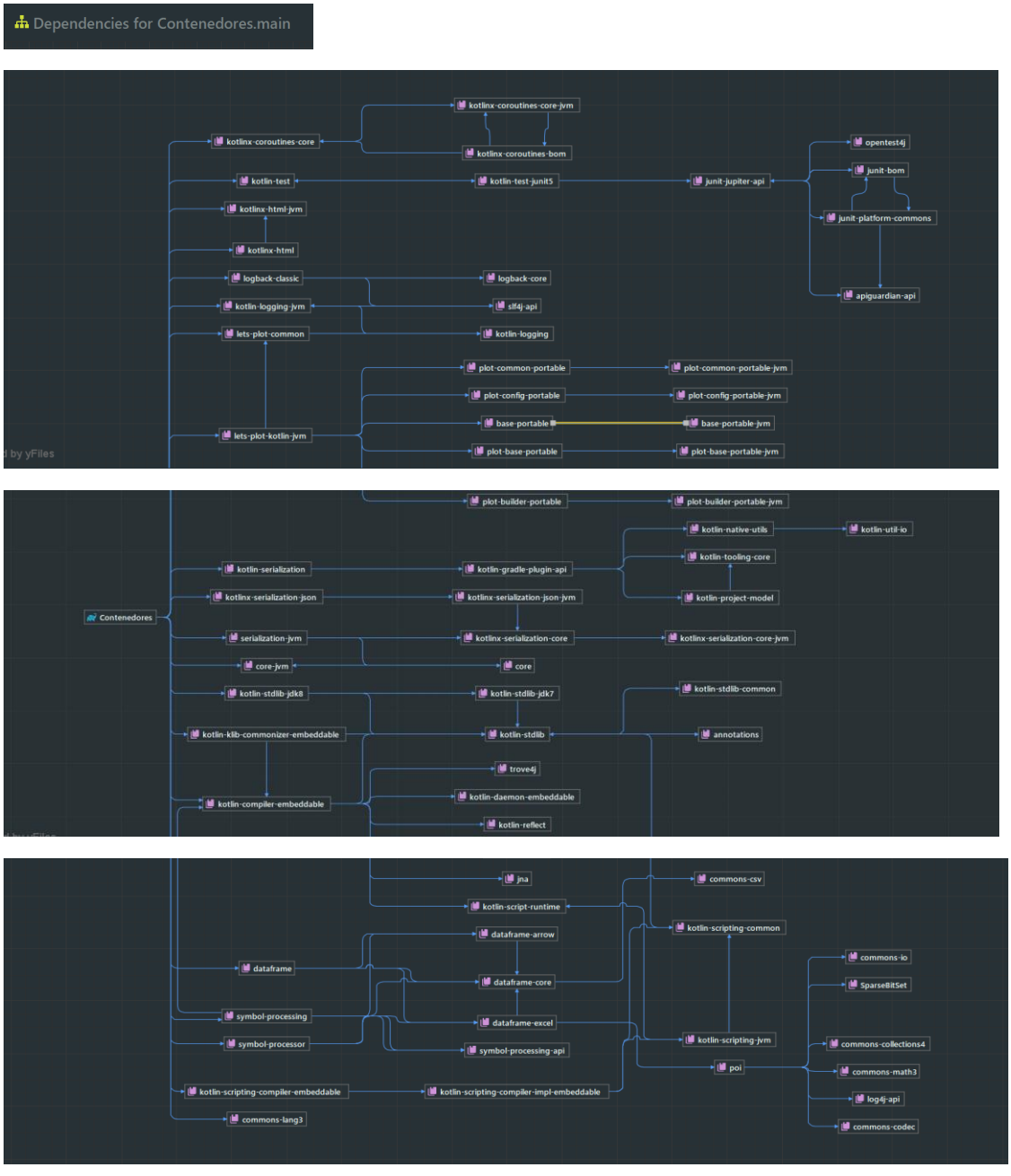
Roberto Blázquez y Francisco Toribio

Contenido

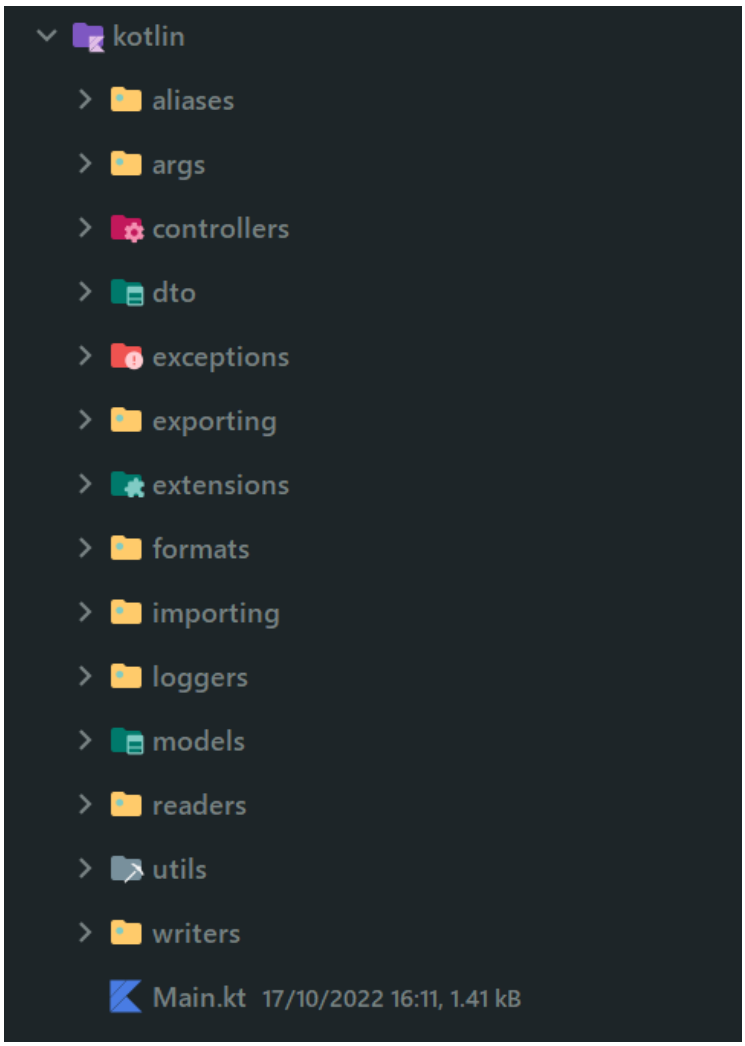
Diseño y propuesta de solución.

DIAGRAMA DE DEPENDENCIAS DE LA APLICACIÓN

Dependencias usadas.



Hemos dividido el código en paquetes para favorecer la modularidad.



Para nuestra propuesta de solución usamos las técnicas aprendidas en los módulos de “Acceso a Datos” y “Programación de Procesos y Servicios”, fomentando su práctica.

Pretendemos optimizar el uso de los recursos del sistema y practicar lo aprendido con mejor o peor éxito. Aprendemos, experimentamos e intentamos que la aplicación sea fiable.

Clases y elementos usados. Justificación tecnológica.

Hemos utilizado:

Lenguaje de programación: Kotlin.

IDE: IntelliJ

Control de versiones: Github

Sistema de control de versiones: GitFlow

Sistema de automatización de construcción de código: Gradle

Typealias para nombrar tipos de datos y estructuras ya existentes en nuestro código de una forma más visual.

Secuencias para optimizar las operaciones sobre colecciones, evitando la creación de objetos temporales entre los pasos de la cadena de computaciones.

DTO's para transportar datos.

También utilizamos el principio de segregación de interfaces para que ninguna clase dependa de métodos que no usa. Además, fomentamos la funcionalidad que nos aporta el uso de interfaces para evitar acoplar nuestro código.

Programación con clases genéricas para generalizar las funciones y poder reutilizarlas en más de una ocasión.

Clases selladas para aprovechar la potencia que nos da que cada subtipo (subclase) de una clase sellada sea otra clase.

Transformación de formatos de la información.

Para la transformación de los datos de formato .csv a formato .json o a formato .xml usamos Kotlin serialization. Realizamos los informes en formato .html.

Realización de las consultas.

Para las consultas usamos DataFrame.

Gráficos.

Para la realización de los gráficos hemos usado los datos filtrados de los csv's con las consultas propuestas en la práctica.

Utilizamos las librerías que Kotlin nos proporciona, Lets-Plot que es una API de Kotlin para crear gráficos de código abierto para datos estadísticos y que tiene ggplot para dibujar los gráficos.

Ejemplo de gráfico:

Aplicación de otras técnicas interesantes.

Inclusión de corrutinas y técnicas de programación asíncrona.

Creación de flujo de trabajo de integración continua (CI) en acciones de GitHub para la comprobación y construcción del proyecto JVM con Gradle.