

ANDREI GARCÍA CUADRA

100405803

MIGUEL HERNÁNDEZ CASSEL

100405956

PRINCIPIOS DESARROLLO SOFTWARE  
NORMATIVA DE CÓDIGO

<b>Memoria 1.....</b>	<b>3</b>
1. Normativa y colectividad de código .....	3



# MEMORIA 1

## 1. NORMATIVA Y COLECTIVIDAD DE CÓDIGO

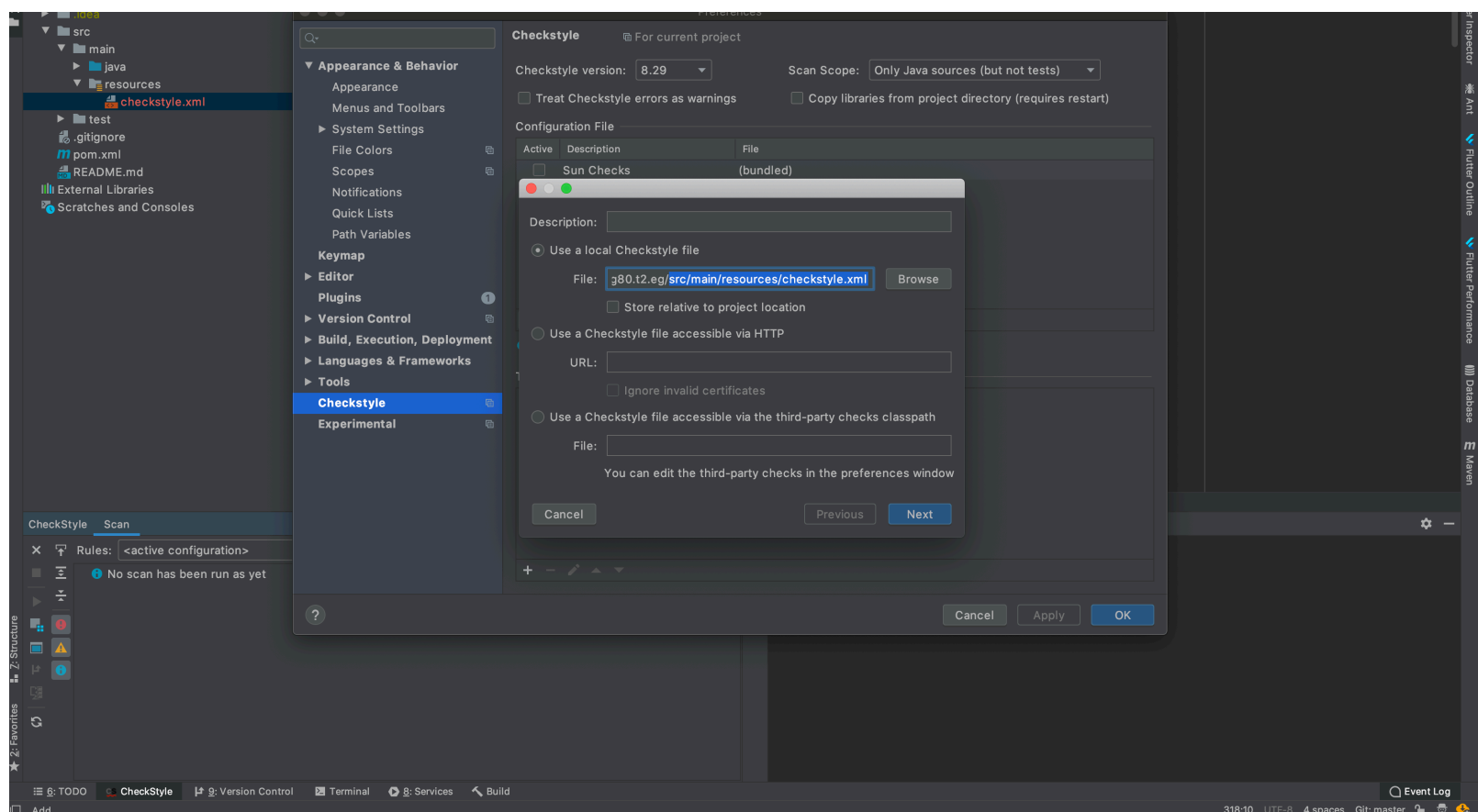
Determinamos el uso de *lint* como estrategia de revisión, estandarización y análisis de código.

Esta elección se debió a la **utilización inter-operativa**, para tener **preconfigurado** el entorno de desarrollo (ya que lint se integra con Maven) y no tener que configurar cada equipo de forma individual (como es el caso de *CheckStyle*).

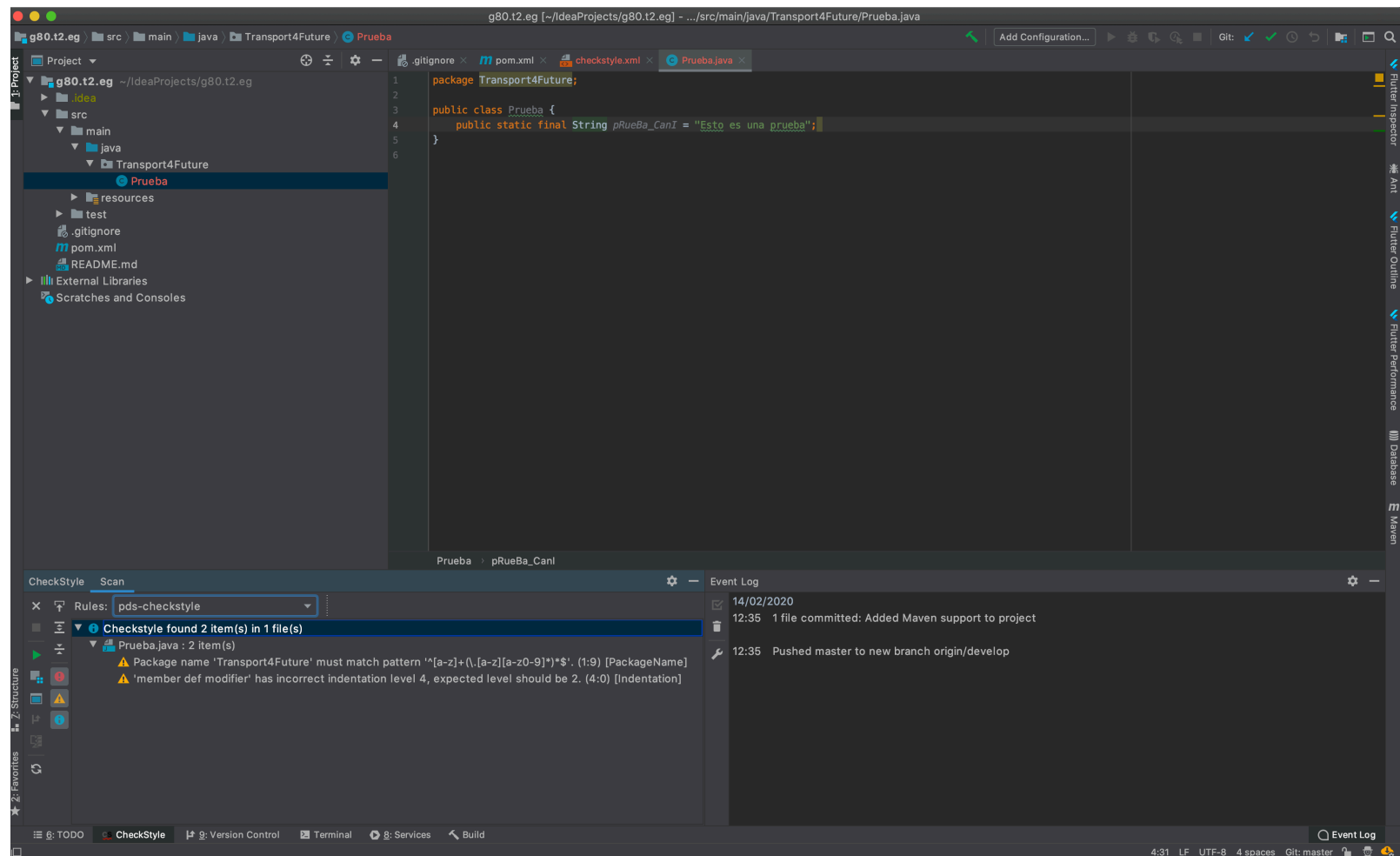
No obstante, esta **tarea fue denegada**, y nos ceñimos al enunciado de la **práctica**, el cual narra **explícitamente** el uso de *CheckStyle*.

Hemos creado un *CheckStyle* propio, el cual puede apreciarse bajo la ruta *src/resources/checkstyle.xml* del repositorio. No podemos documentar todos los estándares, pero sí los importantes. Lo hemos integrado en el *IDE* utilizando el plugin *CheckStyle-Idea*, y agregando la ruta correspondiente al fichero.

Nos hemos **basado** en el *CheckStyle* de *Google* ya que determinamos que se nos adaptaba al completo, editando cualquier **sección** que consideramos y **adaptándola al proyecto**.



*Integración de CheckStyle propio en IntelliJ Idea, utilizando el plugin CheckStyle-Idea.*



*Prueba de uso de CheckStyle, el cual está funcionando correctamente y nos indica.*

## 2.1. Estándar para la organización de archivos

- El nombre de archivo es sensible a mayúsculas y minúsculas.
- El nombre de archivo debe contener la extensión **.java**.
- El nombre de archivo no debe contener espacios ni caracteres especiales.
- Los paquetes deben ser escritos en minúsculas y no contener caracteres especiales.

## 2.2. Estándar para nomenclaturas y variables

- Las **variables** deben ser **CamelCase** capitalization.
- Las **contantes** deben ser letras **mayúsculas** y usar **\_** donde hubiese espacios.

## 2.3. Estándar para métodos y clases

- Los métodos que ejecuten una función determinada (no contamos setters, getters, toString, ...) deben estar documentados con *JDoc*.

## 2.4. Estándar para el tratamiento de excepciones

- Las **excepciones** deben ser **escaladas** y no usar try catch en exceso, es decir, lanzar excepciones desde los métodos.

## 2.5. Estándares extra

- Cada archivo debe incluir la **licencia del proyecto** y no la personal.
- La **importación de paquetes no puede ser wildcard**. Debe ser una importación limpia del paquete en cuestión y no pueden ser importados estáticamente.

## 2.6. Estándar para el diseño y patronización de la aplicación

- La **indentación es libre** y se utiliza la del IDE actual (ya que IntelliJ detecta la indentación actual y la establece como por defecto para todos los refactor).

## 2.7. Estándar para la revisión del código y actualización del repositorio

- **No hay estándares** para esto, pero **sí integración continua**, que realiza esta tarea de forma automatizada.