

Facultad de matemáticas

Estándar de codificación para el conteo de líneas físicas y lógicas

Versión 1.0

Autores:

Aaron Isaac Graniel Arzat

Fernando Joachín Prieto

David Peña Muñoz

Jose Luis Pooc Moo

Andrea Isabel Torres Perez

**CONTROL DE DOCUMENTACIÓN**

**Control de Configuración**

| **Título:** | Estándar de documentación para el conteo de líneas físicas y lógicas |
| --- | --- |
| **Referencia:** | ESP\_EstandarCodificacionProyectoConteoLineas\_v1.0\_2025-02-17 |
| **Autor(es):** | Andrea Isabel Torrez Perez, Aaron Isaac Graniel Arzat |
| **Fecha:** | 17/02/2025 |

**Histórico de versiones**

| **Versión** | **Fecha** | **Estado** | **Responsable** | **Nombre de archivo** |
| --- | --- | --- | --- | --- |
| 1.0 | 17/02/2025 | A | Andrea Isabel Torrez Perez, Aaron Isaac Graniel Arzat | ESP\_EstandarCodificacionProyectoConteoLineas\_v1.0\_2025-02-17 |

**Estado:** (B)orrador, (R)evisión, (A)probado

**Histórico de cambios**

| **Versión** | **Fecha** | **Cambios** |
| --- | --- | --- |
| 1.0 | 17/02/2025 | Ninguna, primera versión. |

**Índice**

[**Introducción 4**](#_4886h27x3gx)

[**Estándar de Codificación 4**](#_5o3db443obby)

[1. Convenciones de Codificación 4](#_4rdr91ym3hr)

[1.1 Nombres de Identificadores 4](#_vrlcx34okt2q)

[1.2 Formato y Estilo 4](#_xbx168nkd65u)

[1.3 Estructura de Código 6](#_3h8f2dfgwbtj)

[2. Convenciones de Documentación Interna 7](#_n3o9cfeszanb)

[2.1 Comentarios en línea 7](#_vwnpueukq2qn)

[2.2 Javadoc 7](#_9ww8sbwcikcb)

# **Introducción**

Este documento define un conjunto de normas y convenciones que deben seguirse para asegurar la uniformidad en la escritura de código del programa encargado de contar las líneas lógicas y físicas de un código fuente, excluyendo comentarios y líneas en blanco.

# **Estándar de Codificación**

## **1. Convenciones de Codificación**

### **1.1 Nombres de Identificadores**

* **Paquetes**: Los nombres de paquetes deben estar en minúsculas y reflejar la organización lógica del proyecto. Ejemplo:

| com.project.module |
| --- |

* **Clases y Enumeraciones**: Deben ser sustantivos en PascalCase. Ejemplo:

| public class CountLines { } |
| --- |

* **Variables**: Deben estar en camelCase. Ejemplo:

| int totalLines; |
| --- |

* **Constantes**: Deben estar en SCREAMING\_SNAKE\_CASE. Ejemplo:

| public static final int MAX\_LINE\_LENGTH = 120; |
| --- |

* **Métodos**: Deben ser verbos en camelCase. Ejemplo:

| public void countLines() { } |
| --- |

* **Booleanos**: Deben comenzar con prefijos como is, has o should. Ejemplo:

| boolean isValid; |
| --- |

### **1.2 Formato y Estilo**

* **Sangría**: Se utilizarán 4 espacios en lugar de tabulaciones.
* **Nombramiento:** Todos los nombres deben escribirse en inglés.
* **Estructuras de código:** Todas las estructuras de código, su cuerpo debe estar entre llaves independientemente de cuántas líneas haya en el cuerpo.
* **Longitud de línea**: Máximo 120 caracteres.

| setText("Long line split"  + "into two parts."); |
| --- |

* **Llaves**: Uso del estilo K&R (egipcio) y siempre se deben usar llaves, independientemente de la cantidad de líneas que tenga la estructura de control.

| if (condition) {  doSomething(); } |
| --- |

* **Importaciones**: Las clases importadas siempre deben enumerarse explícitamente. No se deben utilizar importaciones de comodines (\*),

| import java.util.List; import java.util.ArrayList; import java.util.HashSet; |
| --- |

* **Declaraciones de variables**: No se pueden agregar declaraciones en múltiples líneas. Cada declaración de variable debe escribirse en una sola línea.

| int a; int b; //No permitido  int a; //Permitido  int b; //Permitido |
| --- |

* **Declaraciones de anotaciones**: Las anotaciones deben colocarse en una línea separada inmediatamente antes de la declaración a la que se aplican. No se permite que las anotaciones estén en la misma línea que la declaración.

| @Override  public int method(){}//Correcto  @Override public int method(){} //Incorrecto |
| --- |

### **1.3 Estructura de Código**

* **Condicionales**:

| if (condition) {  doSomething(); } else {  doSomethingElse(); } |
| --- |

* **Bucles**:

Las variables del iterador se pueden llamar i , j , k , etc.

| for (int i = 0; i < total; i++) {  doSomething(i); } |
| --- |

* **Switch**:

| switch (option) {  case 1:  doSomething();  break;  default:  doSomethingElse();  break; } |
| --- |

* **Try-catch:**

| try {  statements; } catch (Exception exception) {  statements; } |
| --- |

## **2. Convenciones de Documentación Interna**

### **2.1 Comentarios en línea**

Se deben usar solo cuando sean necesarios para aclarar código complejo, es decir, cuando aporten claridad en casos donde la intención del código no sea inmediatamente evidente. No se deben usar en código simple. Los comentarios deben estar en el idioma inglés.

| totalLines++; *// Increment the line counter* |
| --- |

### **2.2 Javadoc**

| */\*\*  \* Counts logical and physical lines of source code.*  *\* Skips comments and blank lines.*  *\* @param file Path of the file to process.*  *\* @return Total number of significant lines.*  *\* @throws IllegalArgumentException If total lines <= 0.*  *\*/* public int countLines(String file) throws IllegalArgumentException { } |
| --- |

Cada método debe estar documentado usando Javadoc para garantizar claridad y facilidad de mantenimiento del código.

Se deben incluir una descripción breve pero clara de la funcionalidad del método, así como, etiquetas estándar @param para los parámetros, @return para describir el valor devuelto y @throws si el método lanza excepciones. Además, los comentarios en Javadoc deben estar en inglés.