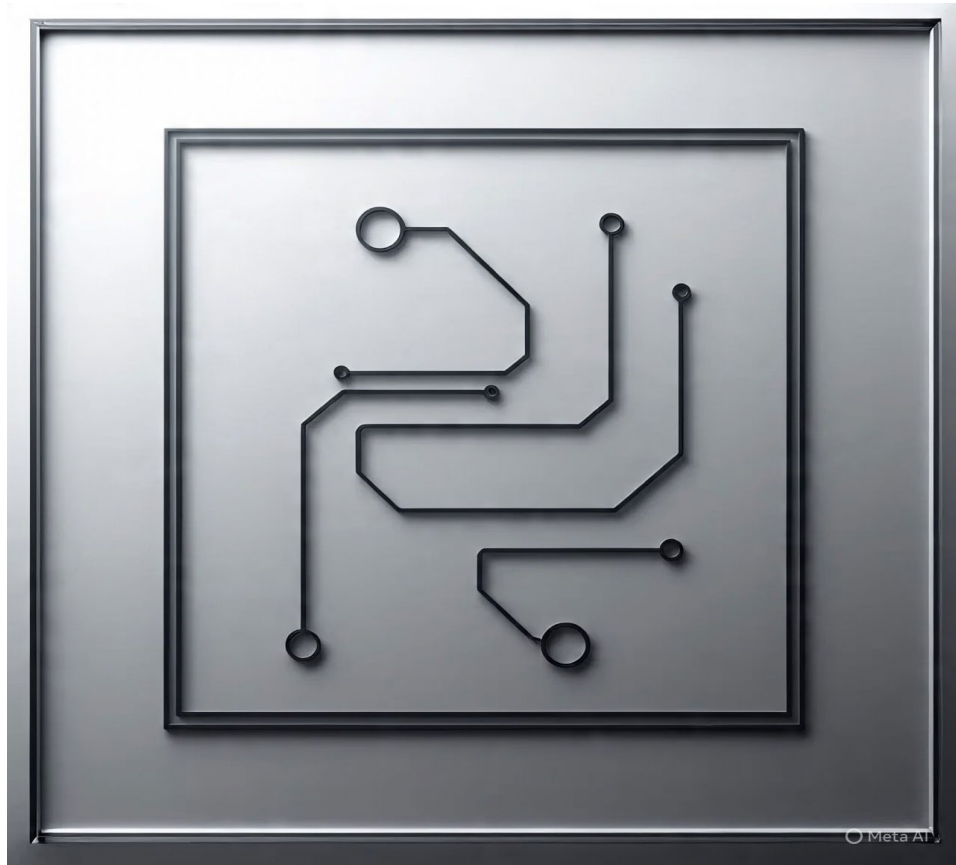


Manual de Documentación del Código

Pensamiento de Programador: Explorando la Lógica de la Programación



Logotipo Logic Programming

Universidad Tecnológica de Pereira
Luis Eduardo Muñoz Guerrero
Agosto 2025

Tabla de Contenido

- Portada
- Tabla de Contenido
- Estructura del Proyecto
- Principales Archivos y Carpetas
- Convenciones de Código
- Ejemplo de Documentación
- Ejemplo Visual de Código
- Bibliografía

Estructura del Proyecto

El proyecto sigue la arquitectura estándar de Angular, con módulos, componentes, servicios y assets bien organizados. La estructura facilita la escalabilidad y el mantenimiento del código.

Estructura de carpetas

```
logic-programming/
├── dist/           # Archivos compilados Angular
├── main.js         # Entrada principal Electron
├── electron-builder.yml # Configuración de empaquetado
├── src/
│   ├── app/       # Componentes, módulos y servicios
│   ├── assets/    # Imágenes y archivos JSON
│   └── ...
└── documentos/   # Documentación y manuales
```

Principales Archivos y Carpetas

- src/app/: Componentes, módulos y servicios principales.
- src/assets/: Imágenes y archivos JSON de ejercicios.
- main.js: Entrada principal de Electron.
- electron-builder.yml: Configuración de empaquetado.
- documentos/: Manuales y documentación del proyecto.

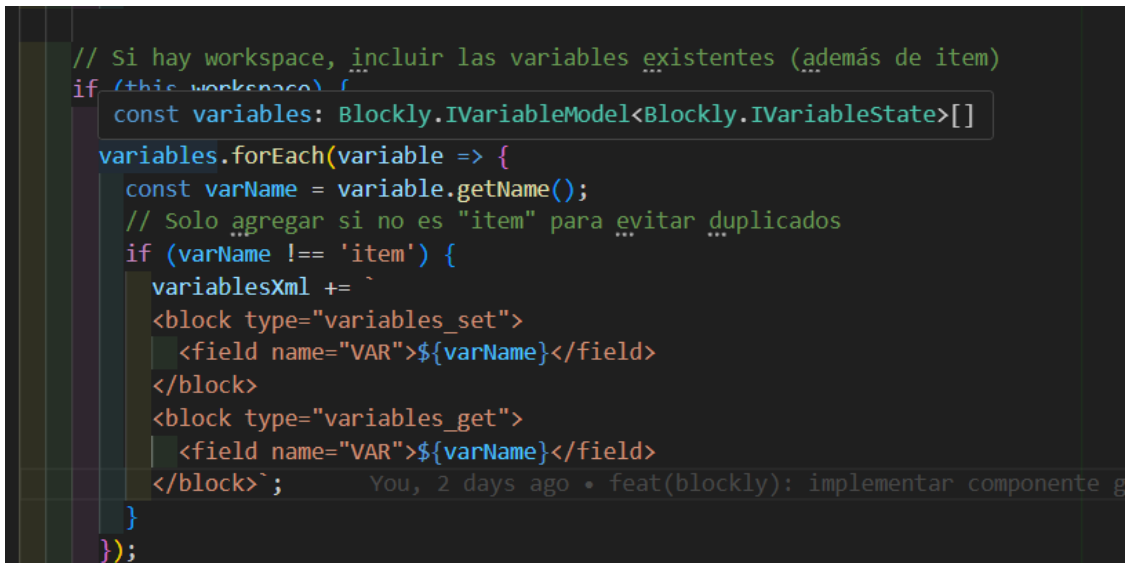
Convenciones de Código

- Uso de TypeScript y Angular 16.
- Componentes y servicios documentados con comentarios JSDoc.
- Nombres descriptivos y en inglés para clases y métodos.
- Separación clara entre lógica de negocio y presentación.
- Uso de interfaces para tipado estricto.
- Buenas prácticas de modularidad y reutilización.

Ejemplo de Documentación

```
/**  
 * Inicializa el workspace de Blockly en el div especificado.  
 * @param element Elemento HTML donde se renderiza Blockly  
 * @returns Instancia de Blockly.WorkspaceSvg  
 */  
initializeWorkspace(element: HTMLElement): Blockly.WorkspaceSvg { ... }
```

Ejemplo Visual de Código



```
// Si hay workspace, incluir las variables existentes (además de item)  
if (this.workspace) {  
  const variables: Blockly.IVariableModel<Blockly.IVariableState>[]  
  variables.forEach(variable => {  
    const varName = variable.getName();  
    // Solo agregar si no es "item" para evitar duplicados  
    if (varName !== 'item') {  
      variablesXml += `  
        <block type="variables_set">  
          <field name="VAR">${varName}</field>  
        </block>  
        <block type="variables_get">  
          <field name="VAR">${varName}</field>  
        </block>`;  
    }  
  });  
}
```

Ejemplo visual de código Blockly

Bibliografía

American Psychological Association. (2020). *Publication manual of the American Psychological Association* (7th ed.). <https://doi.org/10.1037/0000165-000>

Angular. (2025). Documentación oficial. <https://angular.io/>

Electron. (2025). Documentación oficial. <https://www.electronjs.org/docs>

Blockly. (2025). Documentación oficial. <https://developers.google.com/blockly>

Repositorio del proyecto: <https://github.com/Napssters/logic-programming>

Documentación oficial de Electron. (2025). <https://www.electronjs.org/docs>

Documentación oficial de Blockly. (2025). <https://developers.google.com/blockly>

Repositorio del proyecto: <https://github.com/Napssters/logic-programming>