

Programación LOW-CODE

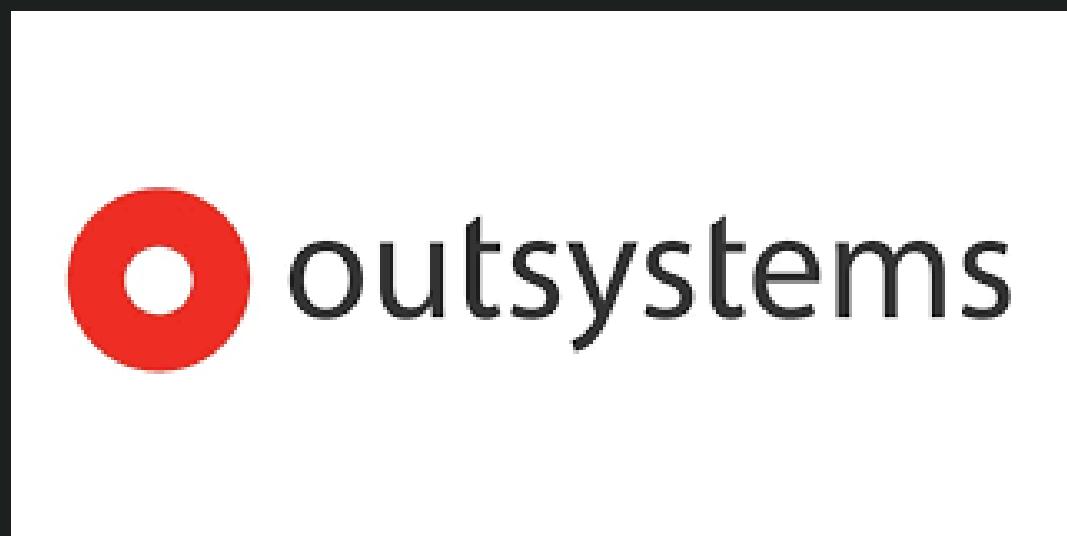
Alberto Ramírez Díaz
Javier Manzano Oliveros
Alejandro Marquez Romero

Teoría - Introducción



Los desarrolladores usan interfaces visuales para diseñar y construir aplicaciones arrastrando y configurando componentes.

Teoría - Primera plataforma low code



2001

80-90

Case: Herramientas para automatizar el desarrollo del software.

4GL: Lenguajes de programación que simplifican tareas comunes, como SQL.

RAD: Modelos como Visual Basic y Delphi que aceleraban la creación de aplicaciones.

Teoría – Evolución

- 2001: Nace OutSystem, considerada la primera plataforma Low Code moderna.
- 2005–2010: Aparecen plataformas como Mendix y Appian.
- 2015–2020: Microsoft Power Apps, Retool y otras herramientas hacen el low code más accesible.
- Hoy: El low code está en auge y se combina con IA.

Teoría - Utilización

- Aplicaciones empresariales-Gestión
- Automatización de procesos y flujos de trabajo-Chatbots
- Aplicaciones para clientes y usuarios finales-Reservas
- Integraciones y conectividad entre sistemas-Conexión con APIs
- Creación de MVPs y prototipos rápidos.

Docker

Image Port(s)

[nodered/nc](#) 1880:1880

The screenshot shows the Docker Desktop interface. On the left, a sidebar menu includes 'Containers' (selected), 'Images', 'Volumes', 'Builds', 'Dev Environments', 'Docker Scout', and 'Extensions'. Below this is an 'Add extensions' button. The main area is titled 'Containers' with a 'Give feedback' link. It displays container statistics: 'Container CPU usage 0.22% / 600%' and 'Container memory usage 17.24MB / 7.56 CB'. A search bar and a 'Only show running' toggle are also present. A table lists eight containers:

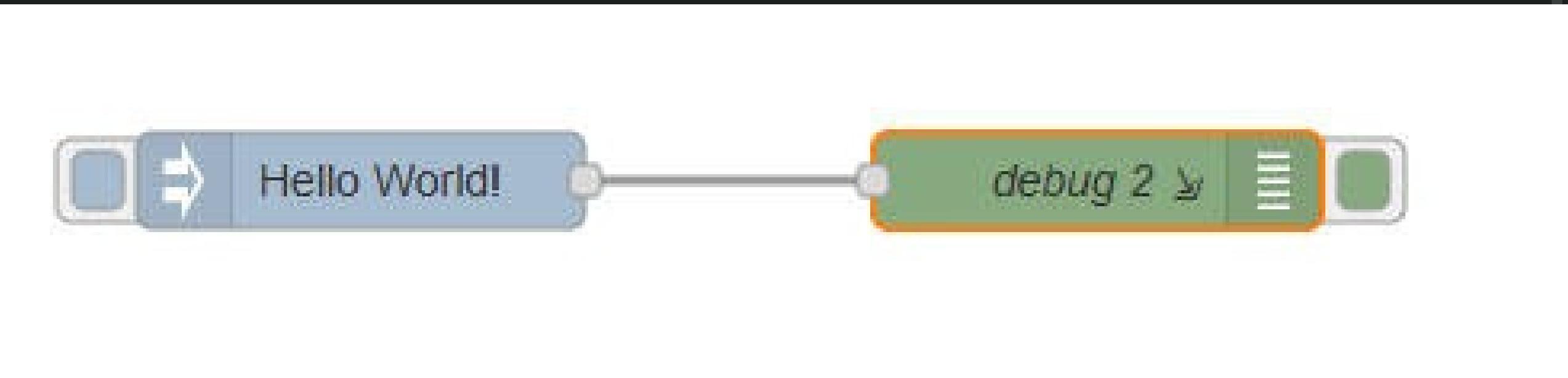
Name	Image	Status	Port(s)	Last started	CPU %	Actions
splashy-whale 43216ff3411	free-willy:latest	Exited (255)	-	1 hour ago	0%	...
barnacle-bob 70398a3bd155	bikini-bottom:latest	Exited	-	2 hours ago	0.22%	...
yarr-matey 87324y65ff32	blue-beard:latest	Exited (255)	-	4 hours ago	0%	...
shrimp-clamtastic 34655h3ulf43	cocktail-sauce:latest	Exited	-	10 hours ago	0%	...
oswald-west 57438h2gff89	cannon-beach:latest	Exited (255)	-	1 day ago	0%	...
endless-summer 5823aa342f51	surfs-up:latest	Exited (255)	-	1 day ago	0%	...

At the bottom, status indicators include 'Docker engine running', 'RAM 4.78 GB', 'CPU 0.41%', 'Disk 50.22 GB avail. of 58.37 GB', 'Connected to Hub', and 'v4.17.0'.

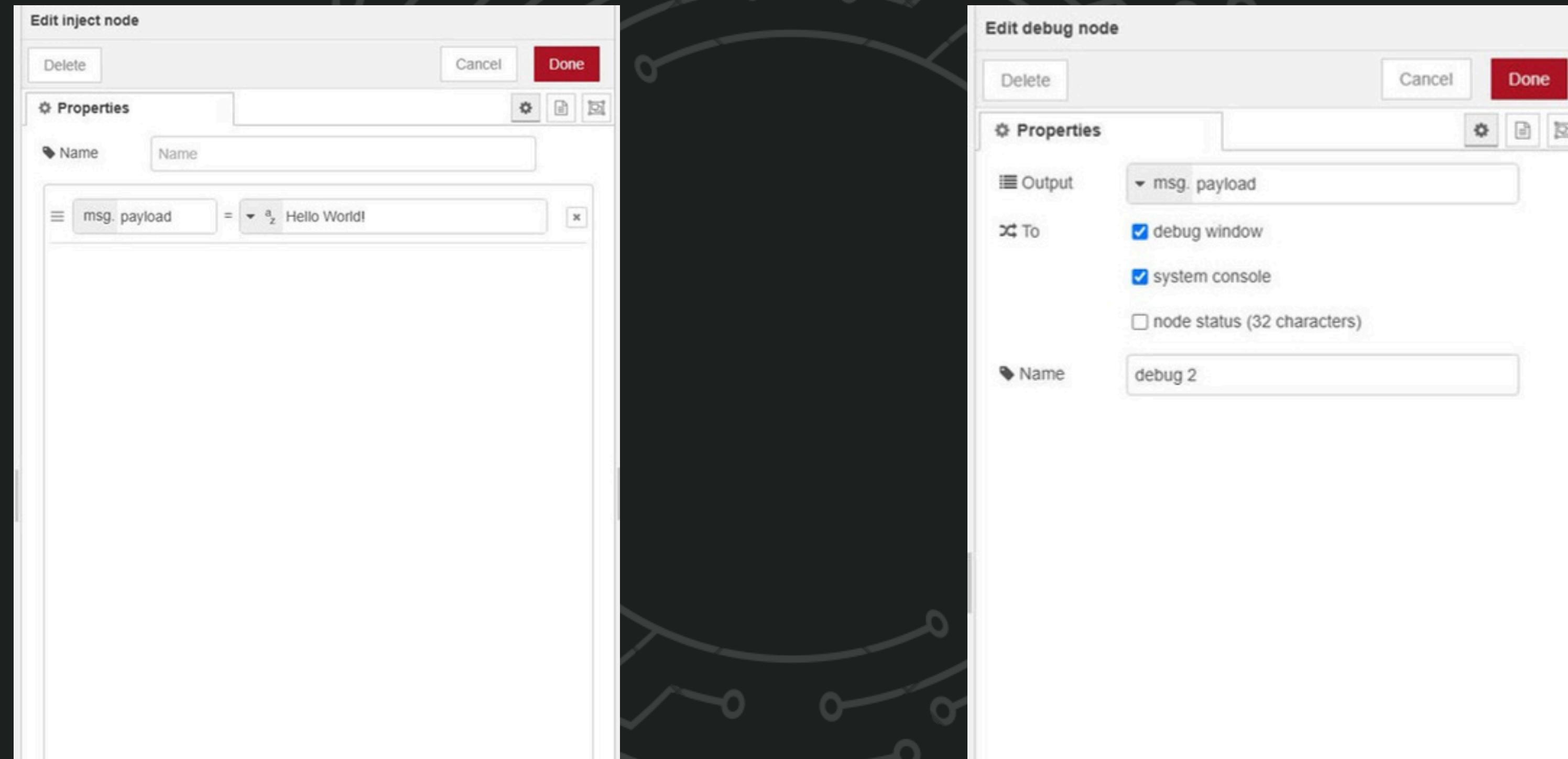
Sintaxis - Introducción

```
public class HelloWorld {  
    public static void main(String[] args) {  
        System.out.println("Hello World");  
    }  
}
```

Sintaxis - Introducción



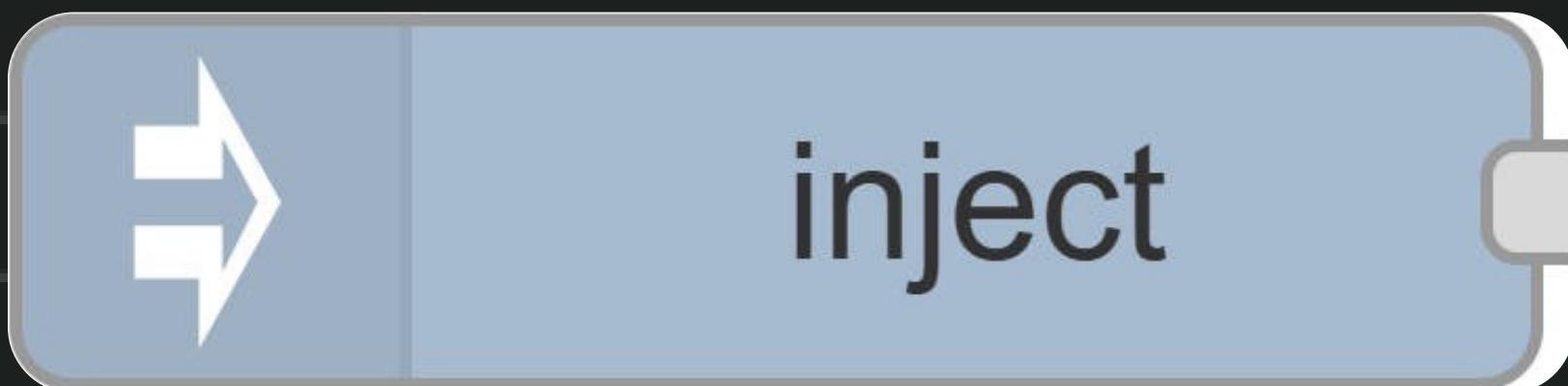
Sintaxis - Introducción



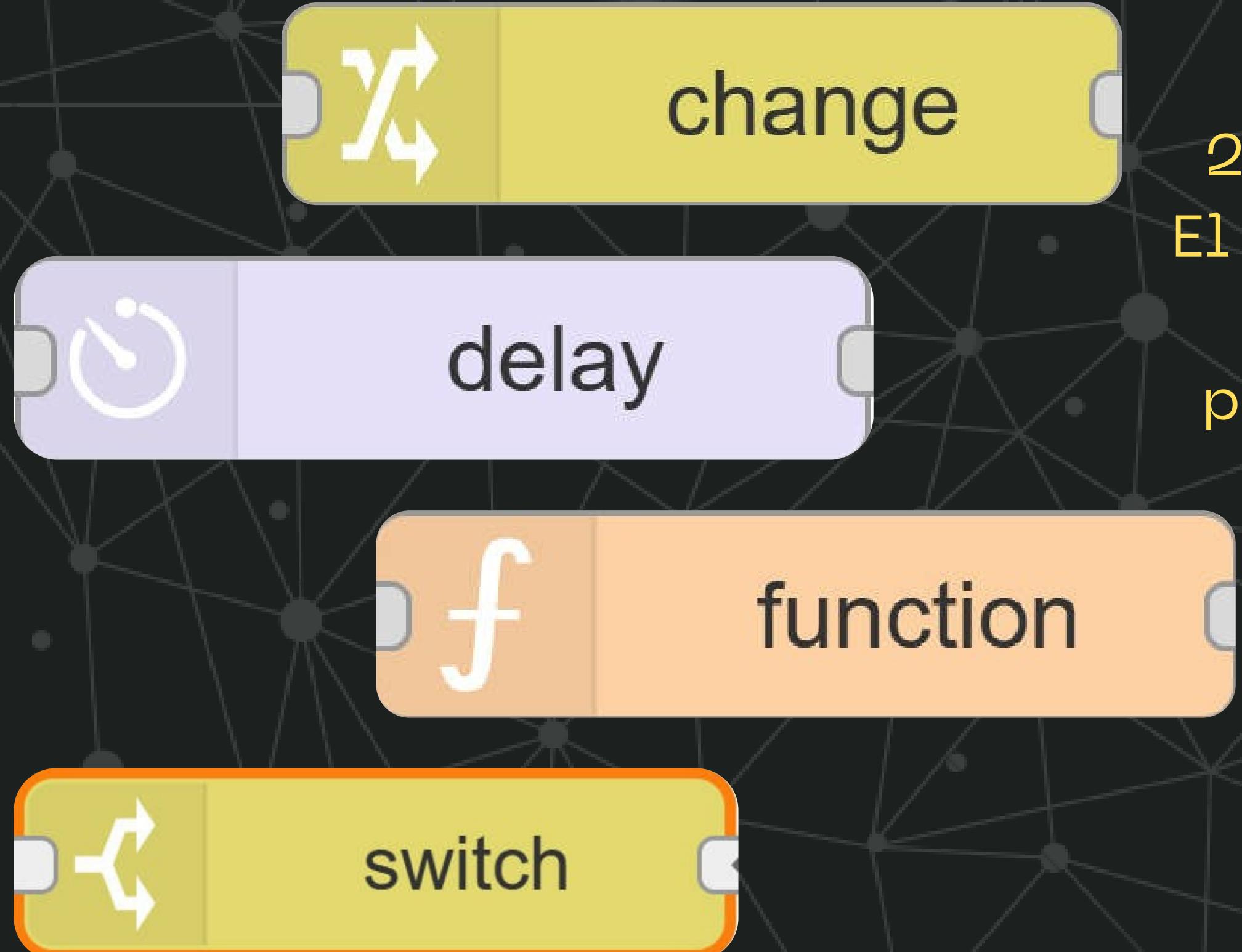
Inject, function, y debug

Orden del flujo paso a paso

1. Entrada (Inject): El flujo comienza cuando el nodo Inject envía el mensaje, como un paquete de datos. Esto simula una entrada, como un sensor o un clic.



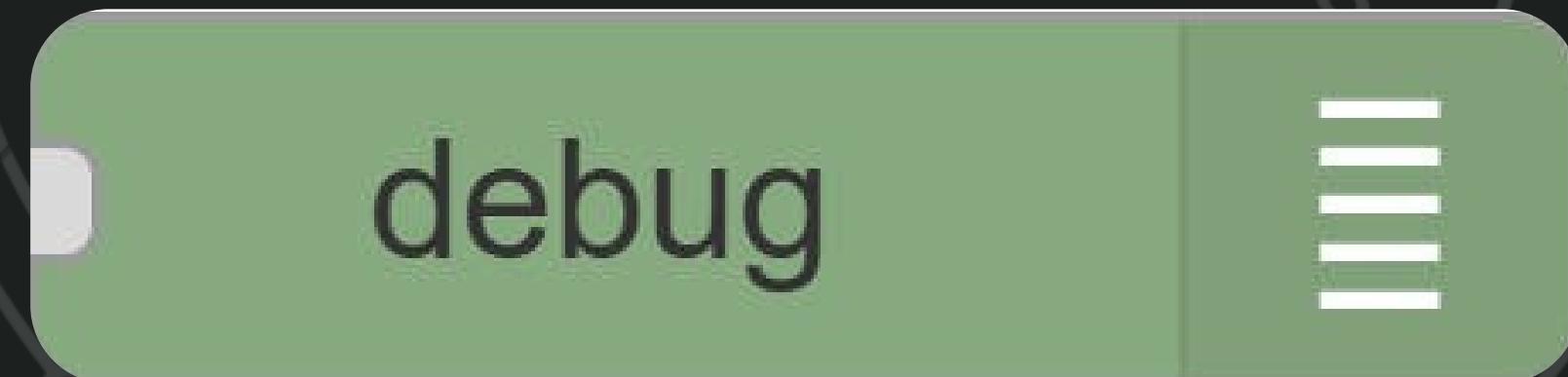
Inject, function, y debug



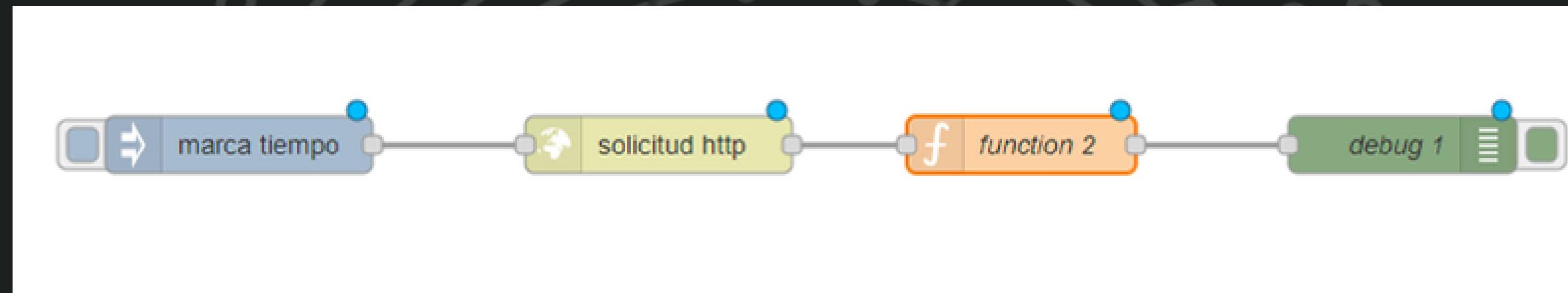
2. Procesamiento (Function):
El mensaje pasa al nodo Function,
que lo modifica. El function
puede ser de muchos tipos pero
modifican el contenido del
paquete datos.

Inject, function, y debug

3. Salida (Debug): Procesa el mensaje final y lo presenta en la pestaña de depuración.

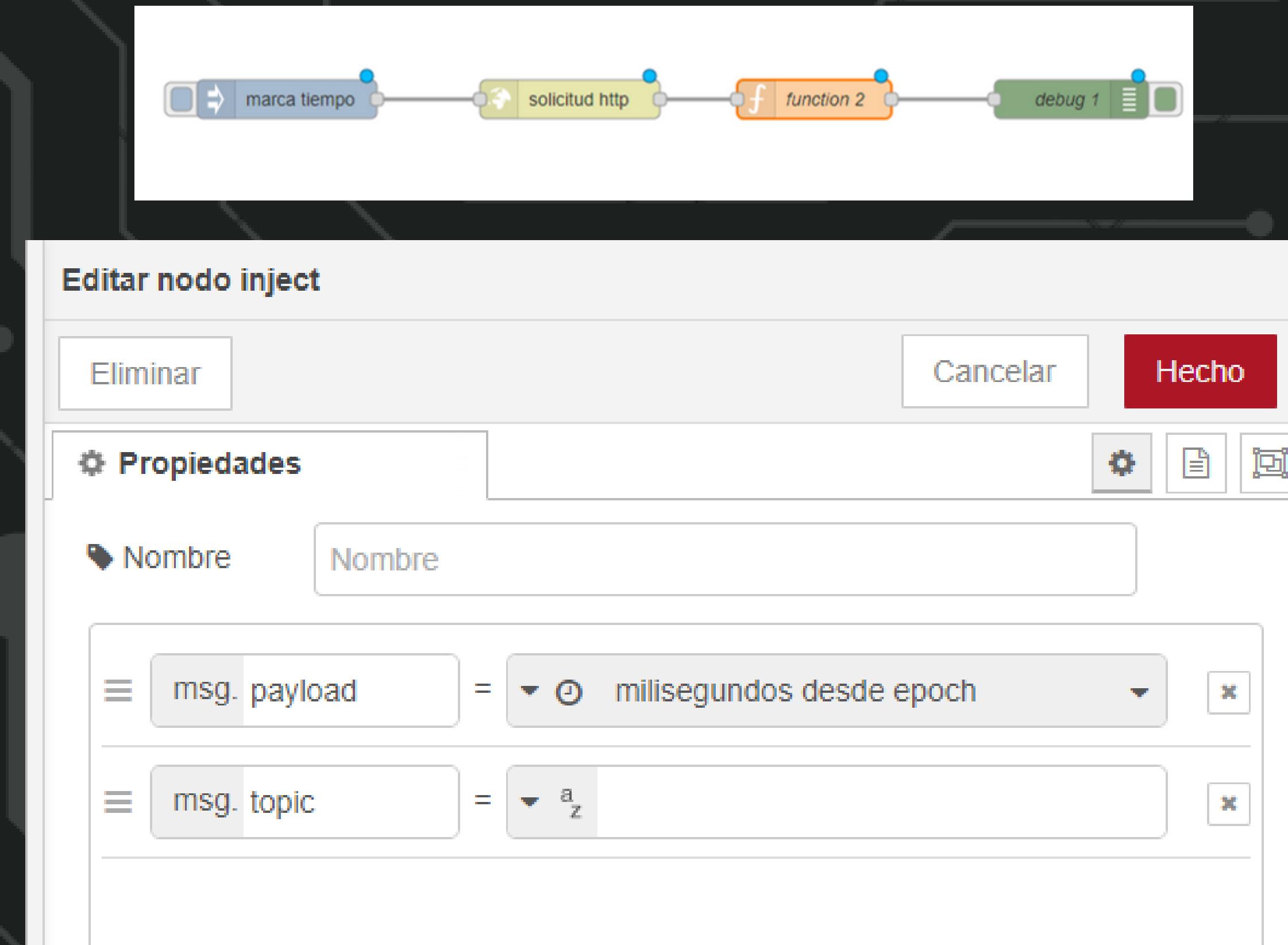


Estructura Básica



Dentro de la estructura básica de un proyecto en node red encontramos el inject, function y el debug. La parte de mayor peso es el function, que es donde se desarrolla el ETL.

Ejemplo Práctico



Ejemplo Práctico

Editar nodo http request

Eliminar Cancelar Hecho

Propiedades

Método: GET

URL: https://api.openweathermap.org/data/2.5/weather?q=Sevilla,ES&aj

Carga: Ignorar

Habilitar conexión segura (SSL/TLS)

Usar autenticación

Habilitar conexión activa (keep-alive)

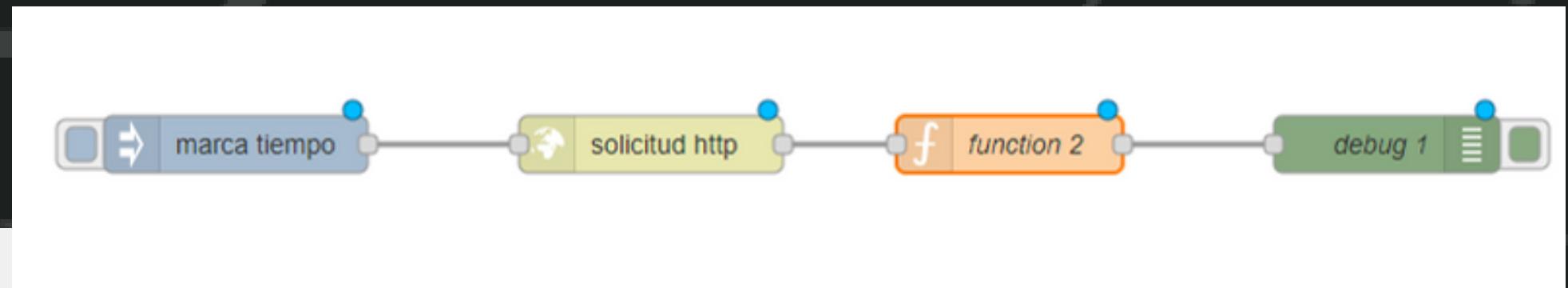
Usar proxy

Enviar solo respuestas que no sean 2xx al nodo Catch

Deshabilitar el análisis estricto de HTTP

Devolver: un objeto JSON analizado

Tip: If the JSON parse fails the fetched string is returned as-is.



EXTRACT

Ejemplo Práctico

marca tiempo → solicitud http → function 2 → debug 1

Editar nodo function

Eliminar Cancelar Hecho

Propiedades

Nombre: function 2

Configuración Al inicio En mensaje Al final

```
1 var data = msg.payload; // Datos de la API
2 var ciudad = data.name; // Nombre de la ciudad
3 var temperatura = data.main.temp; // Temperatura actual
4 var humedad = data.main.humidity; // Humedad
5 var estado = data.weather[0].description; // Estado del clima
6
7 // Construcción del mensaje
8 msg.payload = `El tiempo en ${ciudad} hoy:\n
9   🌡 Temperatura: ${temperatura}°C\n
10  💧 Humedad: ${humedad}%
11  ☀️ Estado: ${estado}`;
12
13 return msg;
```

Ejemplo Práctico

The screenshot shows a Node-RED application interface. At the top, there is a horizontal flow of nodes: 'marca tiempo' (blue), 'solicitud http' (yellow), 'function 2' (orange), and 'debug 1' (green). Below this, a modal dialog box titled 'Editar nodo debug' is displayed. The dialog has buttons for 'Eliminar' (Delete), 'Cancelar' (Cancel), and a prominent red 'Hecho' (Done) button. Inside the dialog, under the 'Propiedades' (Properties) section, there is a 'Salida' (Output) dropdown set to 'msg. payload'. There are three checkboxes: 'ventana depuración' (Debug window) is checked, while 'consola sistema' (System console) and 'estado nodo (32 caracteres)' (Node state (32 characters)) are unchecked. At the bottom, there is a 'Nombre' (Name) field containing 'debug 1'. To the right of the 'Propiedades' section are three small icons: a gear, a document, and a square.

```
graph LR; A[marca tiempo] --> B[solicitud http]; B --> C["function 2"]; C --> D[debug 1]
```

Editar nodo debug

Eliminar Cancelar Hecho

⚙ Propiedades

Salida msg. payload

A ventana depuración
 consola sistema
 estado nodo (32 caracteres)

Nombre debug 1

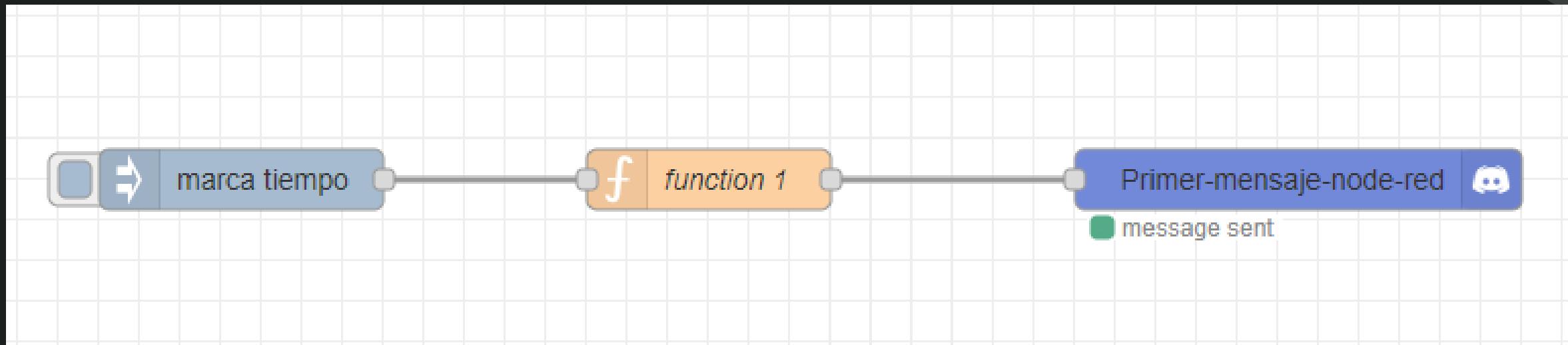
Ejemplo Práctico

The screenshot shows a Node-RED interface with a 'depuración' (debug) tab open. The tab displays the following information:

- Date and time: 17/3/2025, 15:47:12
- Node ID: nodo: debug 1
- Message payload type: msg.payload : string[91]
- Message payload content:
 - El tiempo en Seville hoy:
 - Temperatura: 14.14°C
 - Humedad: 88%
 - Estado: algo de nubes

LOAD

Ejemplo Práctico



Ejemplo Práctico

The screenshot shows a dark-themed messaging application interface. On the left, there's a sidebar with user icons and channel categories like '#notas-recursos', 'CANALES DE TEXTO', 'CANALES DE VOZ', and 'CDP'. The main area displays a channel titled '#planificación-de-sesiones' with a welcome message: '¡Te damos la bienvenida a #planificación-de-sesiones! Aquí empieza el canal #planificación-de-sesiones.' A message from 'n8n APP' is shown: '¡Grande ese Victor! :P'. At the bottom, there's a message input field: 'Enviar mensaje a #planificación-de-sesiones'.

#notas-recursos

CANALES DE TEXTO

deberes-ayuda

planificación-de-s... 2.0

no-relacionado

enlaces

CANALES DE VOZ

HTML

JAVASCRIPT

SALA DE ESTUDIO

JAVA

BASE DE DATOS

CdP

Esda

+ "Manzano" manzano.j... 🌐

EN LÍNEA - 2

"Manzano" manzano.j... 🌐

n8n APP

DESCONECTADO - 3

Javier

NieR

Vladimir

#

¡Te damos la bienvenida a #planificación-de-sesiones!

Aquí empieza el canal #planificación-de-sesiones.

EDITAR CANAL

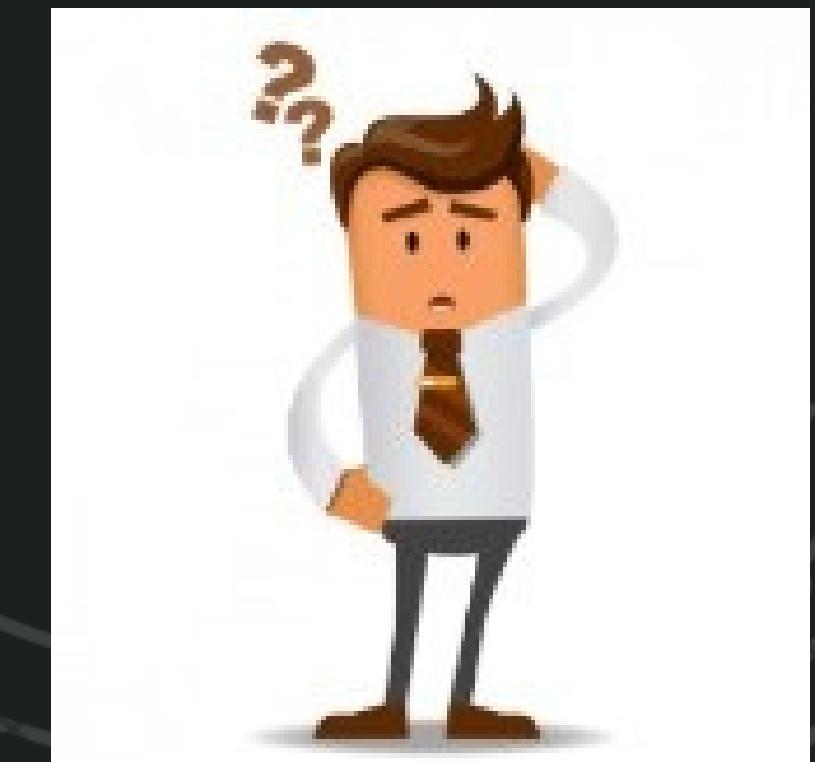
17 de marzo de 2025 NUEVO

n8n APP hoy a las 16:43

¡Grande ese Victor! :P

Enviar mensaje a #planificación-de-sesiones

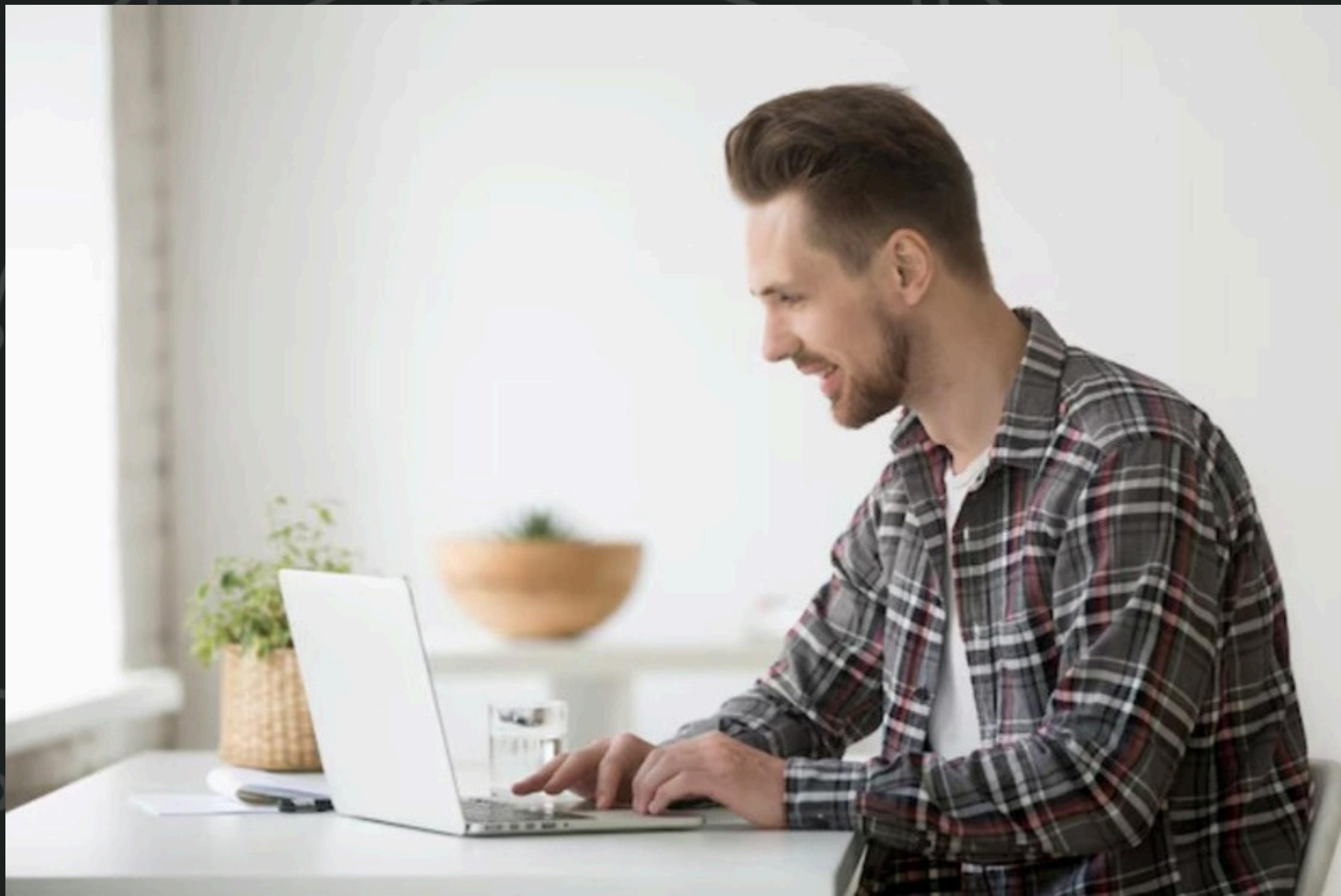
**¿CUÁL ES LA FINALIDAD
DE ESTE NIVEL DE
PROGRAMACIÓN?**



¿PARA QUÉ SE SUELE UTILIZAR?



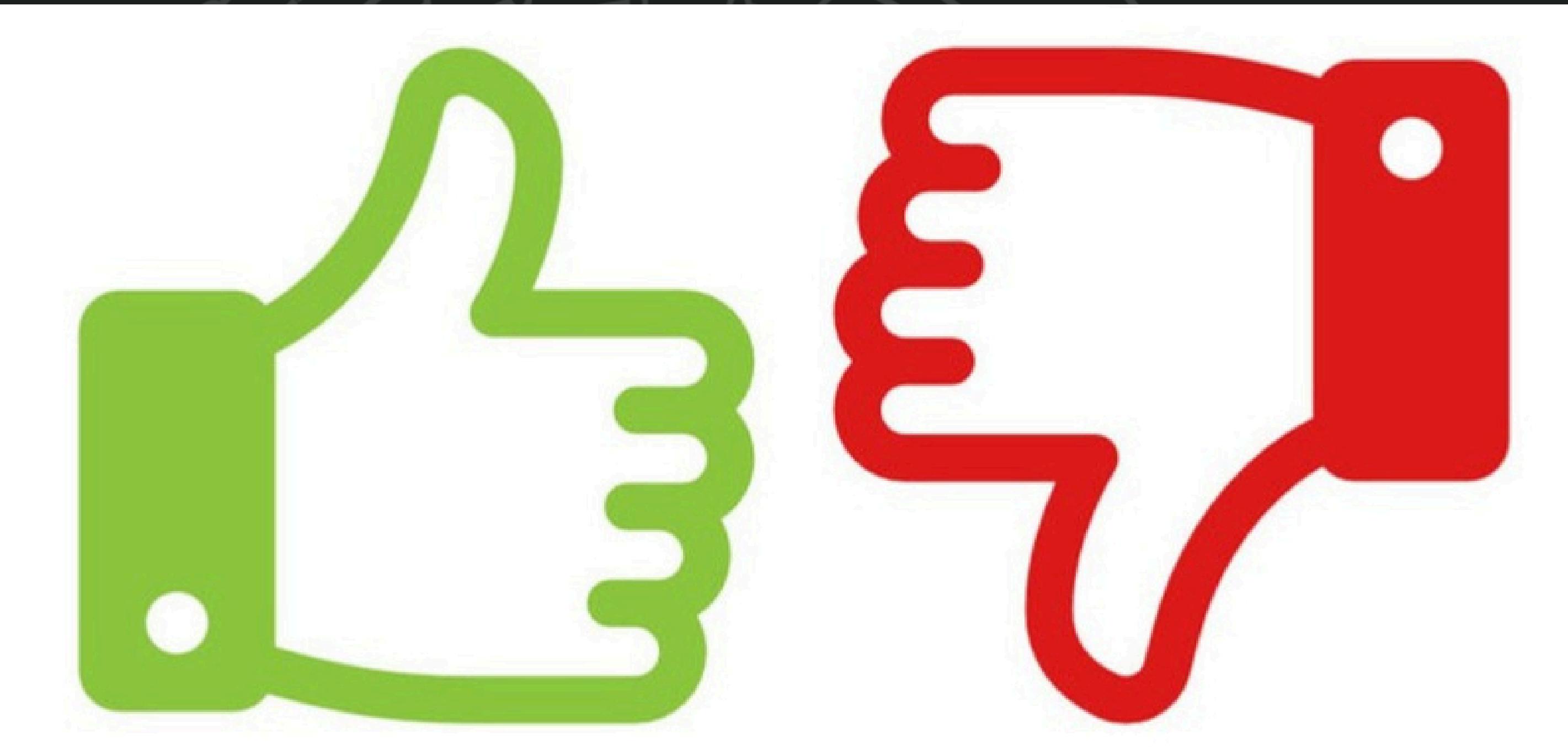
¿Es factible en el uso práctico?



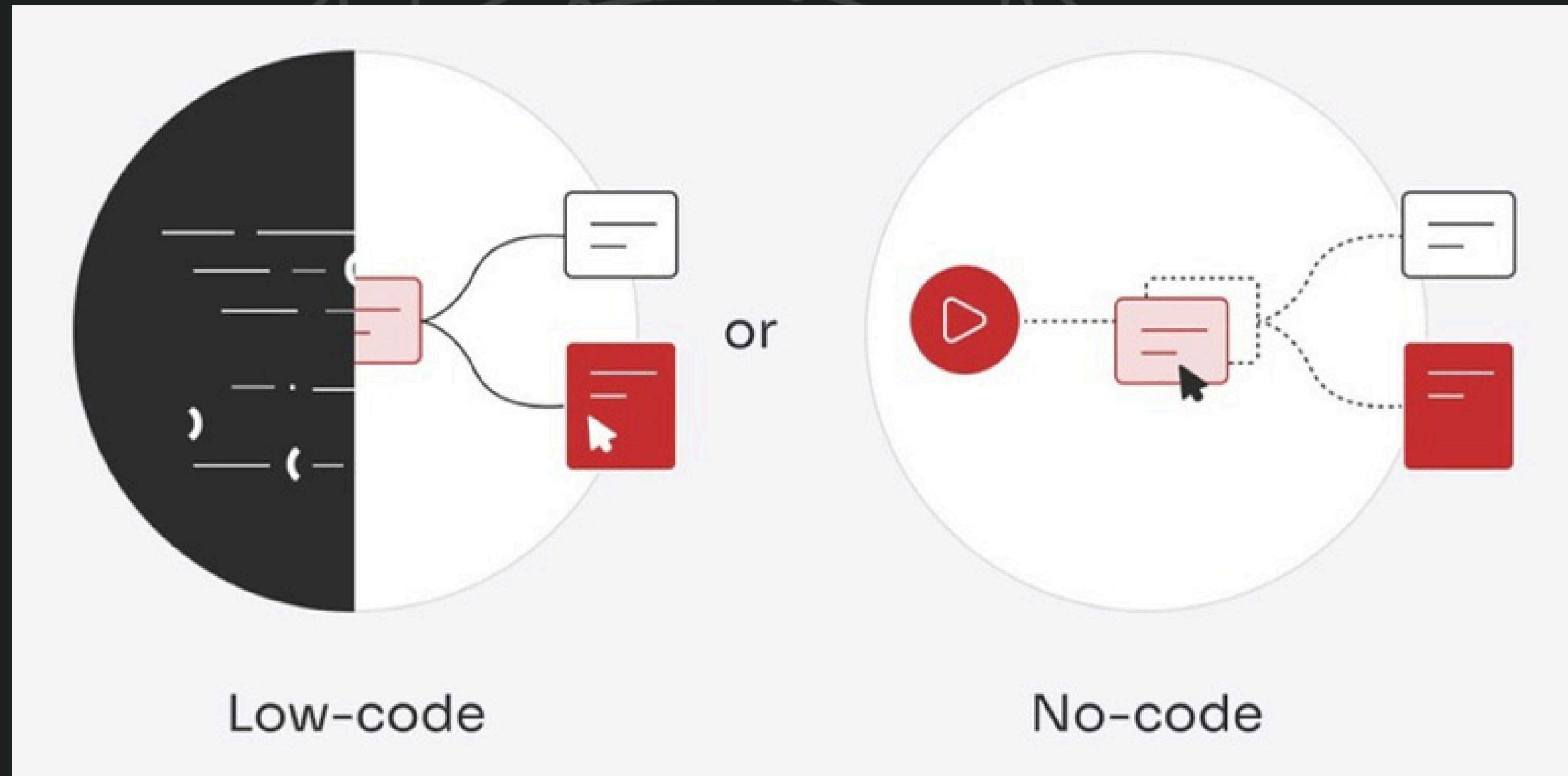
PROS VS CONS SOBRE NODERED



VENTAJAS Y DESVENTAJAS



LOW-CODE VS NO-CODE



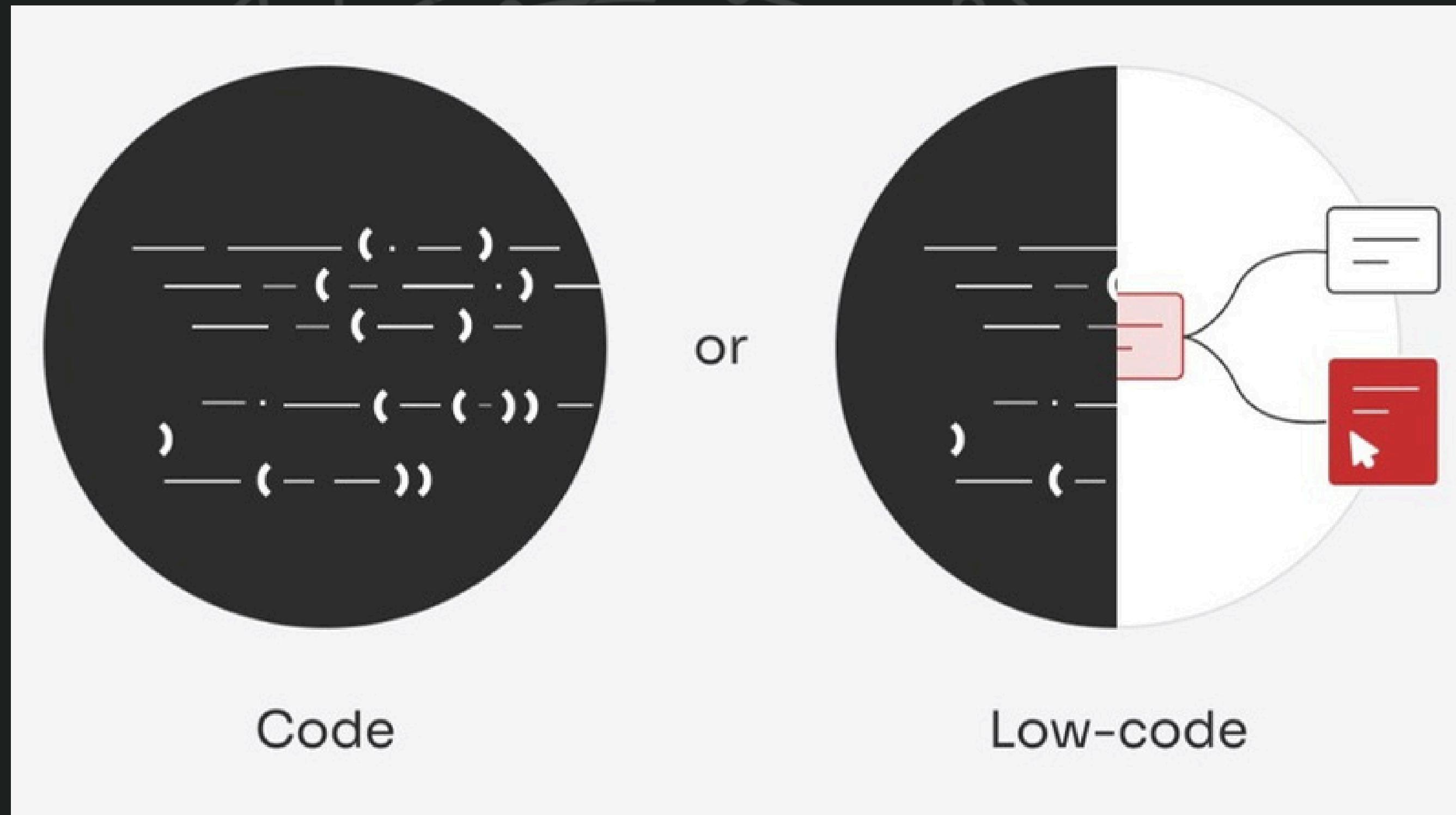
LOWCODE > NOCODE

- > Mayor personalización
- > Más escalable
- > Mejor para programadores con experiencia

NOCODE > LOWCODE

- > Más fácil de aprender
- > Se desarrolla más rápido
- > Mejor para programadores con pocos conocimientos

LOW-CODE VS CODE



LOWCODE > CODE

- > Menor complejidad
- > Más escalable
- > Más rápido de desarrollar

CODE > LOWCODE

- > Mayor control sobre el hardware
- > Mayor rendimiento y optimización
- > Mejor comprensión de los conceptos de programación

¡ESTO HA SIDO TODO!

```
public static void main(String[] args) {
    new Hash3();
}
```