



wavebinder™

Rivoluzione nella gestione
dei dati **front-end**

Indice

1. Il problema dei flussi dati
2. Le sfide dei frontendisti
3. Come funziona Wavebinder™
4. Perché Wavebinder™ funziona

Wavebinder™ è una libreria progettata per aiutare i frontendisti a facilitare la gestione delle dipendenze fra i dati nelle applicazioni front-end.

La crescente domanda di applicazioni web richiede front-end sempre più complessi, che risultano difficili da sviluppare e mantenere.

1.

Il problema dei flussi dati

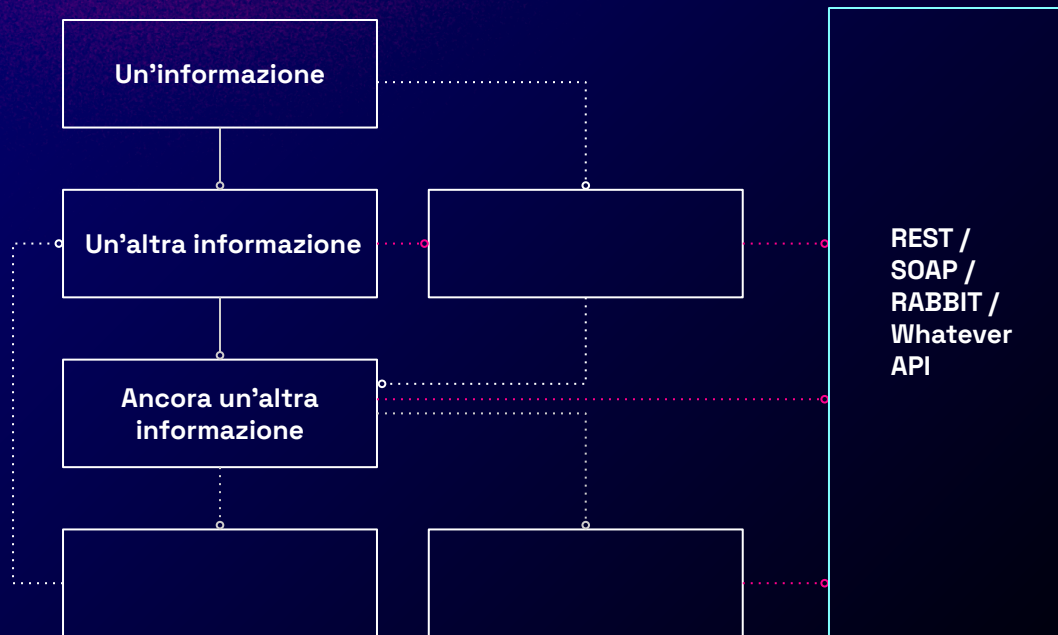
Perché la gestione dei dati è così complessa?

Nelle applicazioni moderne, ogni campo e componente dipende da altri dati.

I dati non si aggiornano in modo lineare.

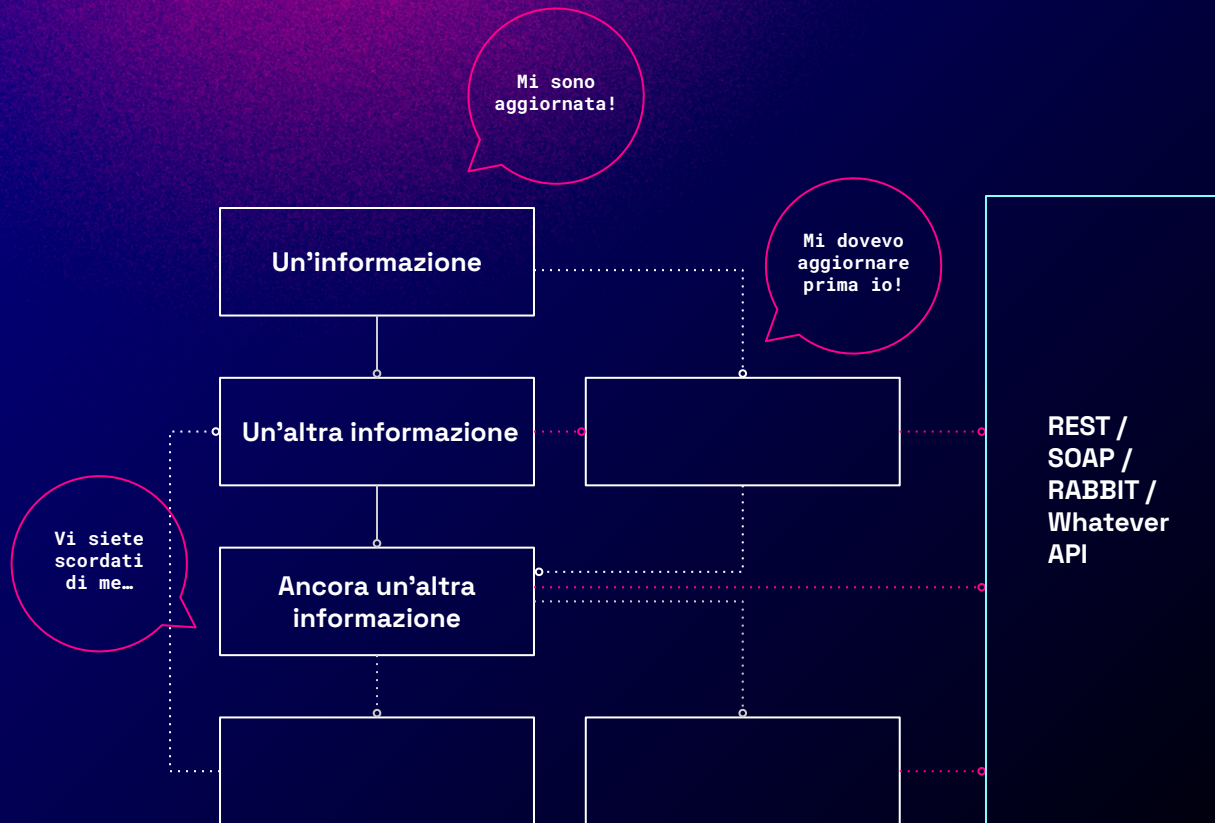
Le complessità aumentano quando si utilizzano API lente e asincrone.

Le interfacce sono sempre più complesse e dinamiche.



Quali sono le conseguenze?

1. Tempi lunghi e ritardi nello sviluppo delle interfacce
2. Errori e comportamenti indesiderati
3. Esperienza utente non ottimale
4. Difficoltà nel mantenimento e dell'aggiornamento delle logiche



2.

Le sfide dei frontendisti

Cosa cercano gli sviluppatori front-end?

Gli sviluppatori front-end cercano strumenti che semplifichino la loro vita e ottimizzino il flusso di dati tra componenti.

La gestione manuale delle dipendenze dati diventa insostenibile in applicazioni complesse.



Qual è la risposta ai loro bisogni?

WAVEBINDER™ è stato sviluppato per rispondere a queste esigenze, offrendo automazione, flessibilità e controllo.



Wavebinder™

WAVEBINDER™ offre una compatibilità e una facilità d'uso che lo rendono ideale per l'integrazione nei tuoi progetti front-end.

Ecco alcune delle sue caratteristiche principali:

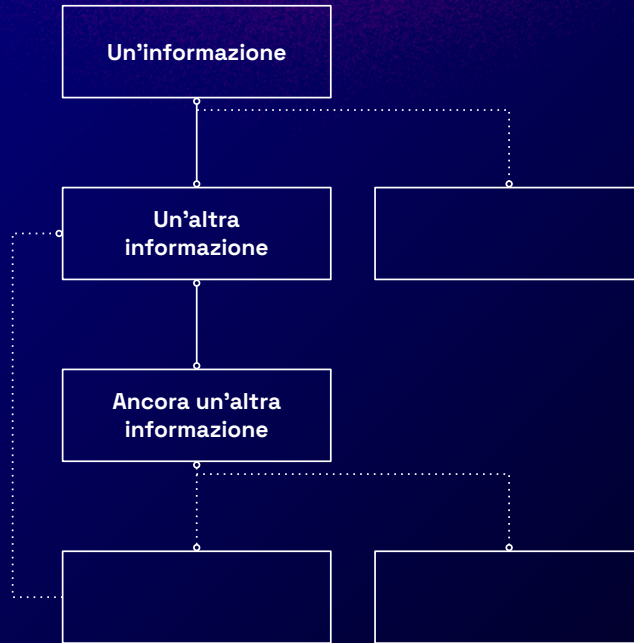
- Disponibile come pacchetto NPM
- Basato su RXJS
- Funziona perfettamente con i principali framework:
 - Vue.js
 - Angular
 - React



3.

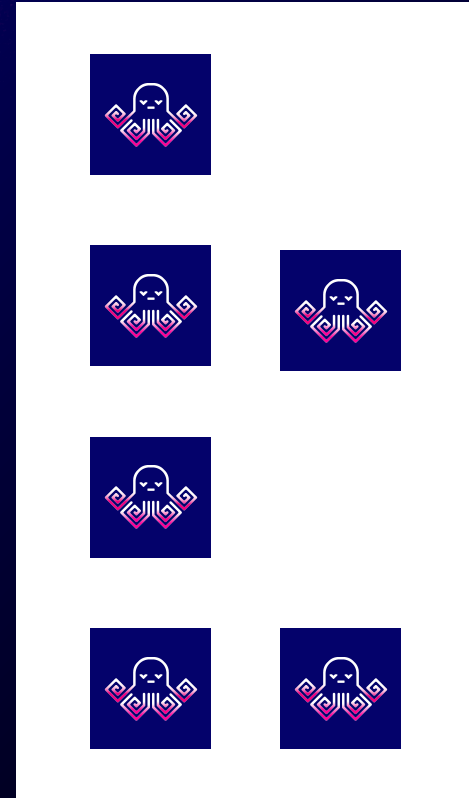
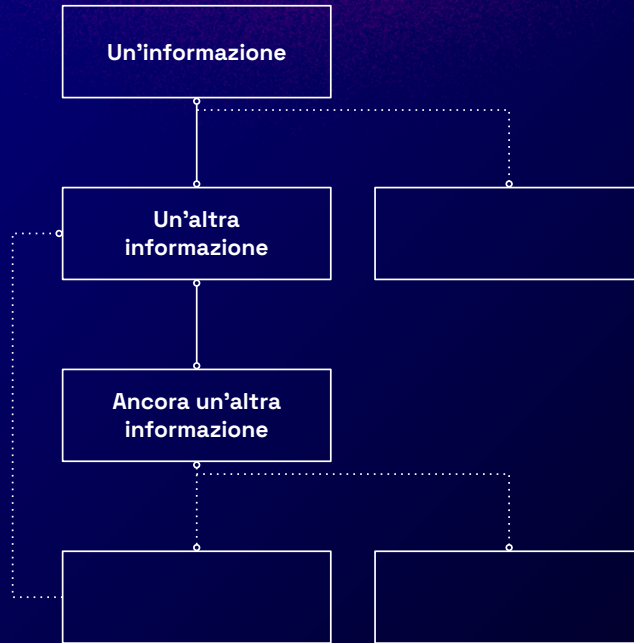
Come funziona Wavebinder™

Modellazione dei dati come nodi di un grafo



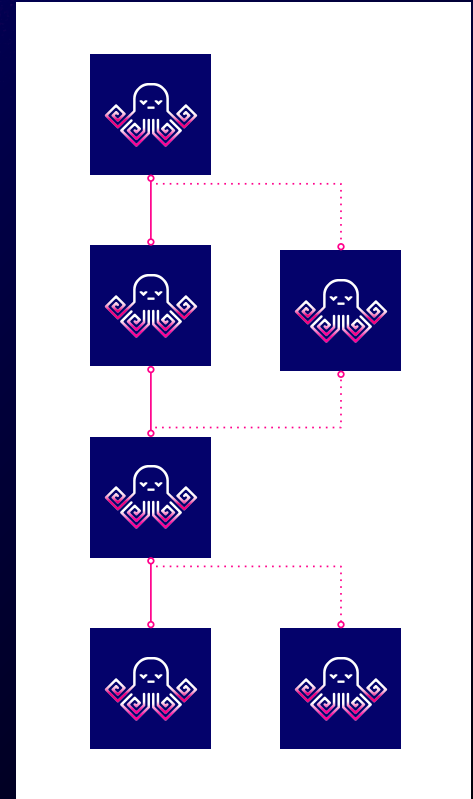
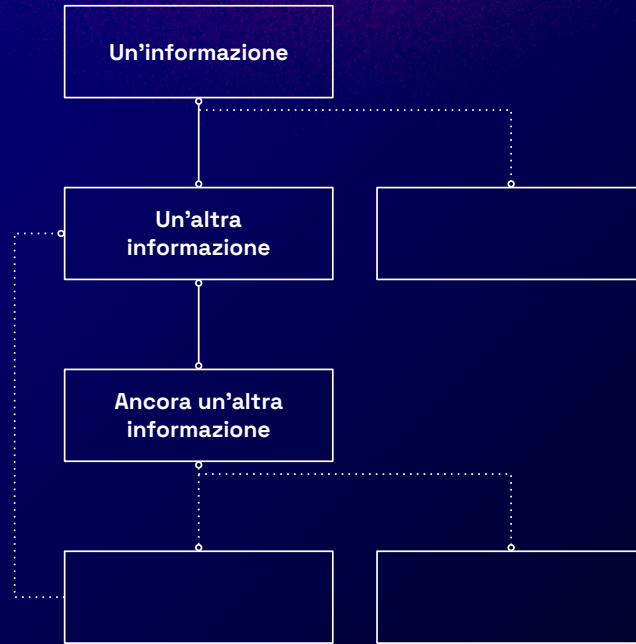
Definizione dei nodi in un oggetto JSON

```
const config: =  
[  
  {  
    "name": "region",  
    "type": "MULTI",  
    "path": "/region",  
    "la": {  
      "type": "GET",  
      "addr": "/region/list",  
      "serviceName": "RETRIEVE_DATA"  
    },  
    "dep": []  
  },  
  {  
    "name": "province",  
    "type": "MULTI",  
    "path": "/province",  
    "la": {  
      "type": "GET",  
      "addr": "/{region}/province/list",  
      "serviceName": "RETRIEVE_DATA"  
    }  
  },  
  ...  
]
```



Dipendenze tra i nodi nella struttura JSON

```
{  
  "name": "city",  
  "type": "MULTI",  
  "path": "/city",  
  "la": {  
    "type": "GET",  
    "addr": "{region}/{province}/city/list",  
    "serviceName": "RETRIEVE_DATA"  
  },  
  "dep": [  
    {  
      "nodeName": "region",  
      "isOptional": false,  
      "onUpdate": true,  
      "type": "PATH_VARIABLE"  
    },  
    {  
      "nodeName": "province",  
      "isOptional": false,  
      "onUpdate": true,  
      "type": "PATH_VARIABLE"  
    }  
  ]  
}
```



Specificare le diverse proprietà per ogni nodo

Il tipo di dato che contiene
(es. un singolo valore,
una lista, un oggetto, ...)

L'azione necessaria per
caricare il suo valore (es. una
chiamata API, una selezione da
parte dell'utente, ...)

```
const config: =  
[  
  {  
    "name": "region",  
    "type": "MULTI",  
    "path": "/region",  
    "la": {  
      "type": "GET",  
      "addr": "/region/list",  
      "serviceName": "RETRIEVE_DATA"  
    },  
    "dep": []  
  },  
  {
```

Il percorso in cui
memorizzarlo su
un modello


Le sue dipendenze
da altri nodi

Specificare le diverse proprietà per ogni nodo

Il tipo di dato che contiene
(es. un singolo valore,
una lista, un oggetto, ...)

L'azione necessaria per
caricare il suo valore (es. una
chiamata API, una selezione da
parte dell'utente, ...)

```
const config: =  
[  
  {  
    "name": "region",  
    "type": "MULTI",  
    "path": "region",  
    "action": "GET",  
    "dep": ["region", "WAVE_DATA"]  
  },  
  {  
    "name": "wave_data",  
    "type": "MULTI",  
    "path": "wave_data",  
    "action": "POST",  
    "dep": ["wave_data"]  
  }  
]
```

A stylized white and pink character with a speech bubble and a code editor background. The character has a white outline with pink-to-white gradient fill on its body and arms. It has a speech bubble above it containing the text: "Abbiamo sviluppato un tool CLI che lo renderà semplicissimo." The background is a dark blue code editor with a light blue border, showing a JSON configuration for a CLI tool. The code includes fields for name, type, path, action, and dependencies (dep).

Abbiamo sviluppato
un tool CLI
che lo renderà
semplicissimo.

Il percorso in cui
memorizzarlo su
un modello

Le sue dipendenze
da altri nodi

COME FUNZIONA WAVEBINDER™

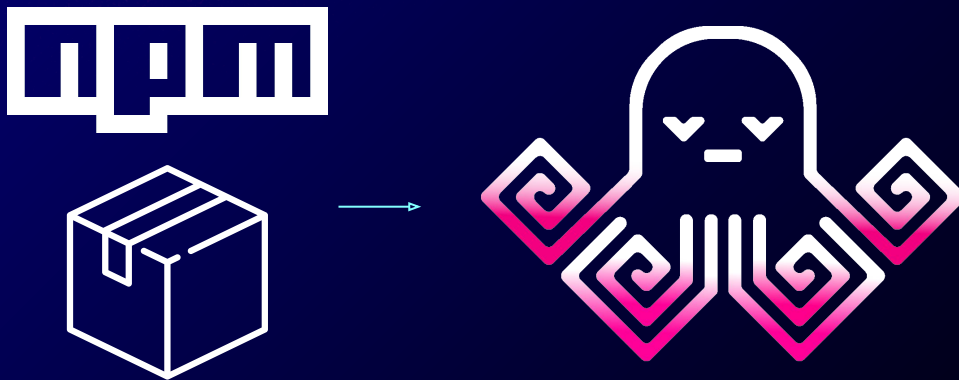
Installazione e configurazione facili

WAVEBINDER™ si integra facilmente nei progetti esistenti.

Basta eseguire «npm install wave-binder» per iniziare.

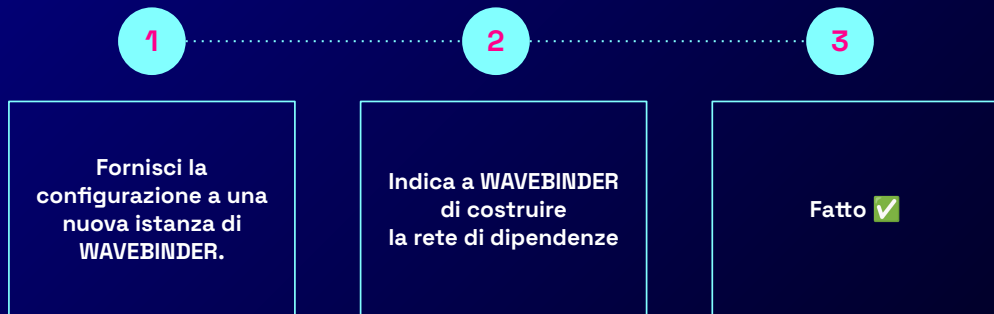
Con pochi passaggi, puoi configurare i nodi e le loro dipendenze usando JSON.

Anche per gli sviluppatori alle prime armi con l'uso di nodi, il tool CLI semplifica la creazione e la gestione delle dipendenze dati.



Funzionalità

Una volta installato **WAVEBINDER™** e modellato il tuo problema nella struttura di configurazione, sei quasi pronto per partire.



```
const wb: WaveBinder = new WaveBinder(config);
```

```
wb.tangleNodes();
```

Funzionalità

WAVEBINDER™ espone:

- I metodi necessari per recuperare i nodi

```
const listNode = wb.getNodeByName('myListNode');  
listNode.next(3);
```

```
const regionNode = wb.getNodeByName('region');  
regionNode.setSelection(1);
```

```
const regionListNode = wb.getNodeByNameAndType('regionList', 'LIST');  
regionListNode.children[1].setSelection(1);
```

```
public constructor(protoNodes: ProtoNode[],  
                  extApis: Map<string, HttpServiceSetting>,  
                  customFunctions: FunctionInstance[]) {  
  }  
  
public tangleNodes(): void {  
  }  
}
```


Funzionalità

WAVEBINDER™ espone:

- I metodi necessari per recuperare i nodi
- Vari modi per interagire con essi.

```
}  
  
public getDataPool(): void {  
}  
  
public getNodesInfo(logDepth?: number): WaveBinderNodeView[] {  
}  
  
public getNodeByName(name: string): WaveBinderNode {  
}  
  
public getNodeByNameAndType(name: string,  
                             type: string): WaveBinderNode {  
}  
  
public getNodeByNameAmongNodes(name: string,  
                                nodes: WaveBinderNode[])  
    : WaveBinderNode {  
}  
  
public getNodeByNameAndTypeAmongNodes(name: string,  
                                       type: string,  
                                       nodes: WaveBinderNode[])  
    : WaveBinderNode {  
}
```

COME FUNZIONA WAVEBINDER™

Documentazione

Ogni aspetto della configurazione e gestione di **WAVEBINDER™** è documentato su www.wavebinder.it.

Questo assicura agli sviluppatori una guida costante per configurazioni avanzate e casi particolari, rendendo l'adozione ancora più semplice.

Why choose Wavebinder?

Data relationships are messy, APIs are slow, and dependencies make everything harder. WaveBinder helps you manage it all with ease.

Data as Graph Nodes

Model data points as nodes, clearly defining dependencies to simplify even the most complex relationships.



Powered by RXJS

Built on top of RxJS, ensuring robust asynchronous data handling and state management.



Multi-framework Support

Seamlessly integrates with your favorite frameworks like React, Vue, and Angular.



DOCUMENTATION →

BROCHURE →

EXAMPLE#1 →

The screenshot shows the documentation page for the `IterationObj` interface. The left sidebar contains a navigation tree for the `wave-binder` package, with `IterationObj` selected. The main content area displays the following information:

- Interface IterationObj**
Represents an object used to track iteration data during certain operations.
- Defined in** `src/bodynode-def.ts:128`
- Index**
- Properties**
 - `fatherName`
 - `index`
- Properties**
 - fatherName**
`fatherName: string`
The name of the parent node or context.
Defined in `src/bodynode-def.ts:128`
- index**
 - `index: number`
The index in the current iteration.
Defined in `src/bodynode-def.ts:128`

On the right side, there are navigation links for **Settings**, **On This Page**, **Properties**, `fatherName`, and `index`.

4.

Perché Wavebinder™ funziona

Un libreria potente, con anni di
esperienza, che ha la visione di astrarre
la complessità una volta per tutte.

Permette a chiunque di concentrarsi solo su ciò che il cliente
desidera davvero: business logic.

Ordine e precisione: dipendenze e azioni che organizzano il tuo codice

Per ogni nodo, ti viene chiesto di specificare due cose molto importanti:

- Dipendenze
- Azione di caricamento

```
"dep": [  
  {  
    "nodeName": "region",  
    "isOptional": false,  
    "onUpdate": true,  
    "type": "PATH_VARIABLE"  
  },  
  {  
    "nodeName": "province",  
    "isOptional": false,  
    "onUpdate": true,  
    "type": "PATH_VARIABLE"  
  }  
]
```

Le dipendenze sono semplicemente riferimenti ad altri nodi

```
"la": {  
  "type": "GET",  
  "adr": "/region/list",  
  "serviceName": "RETRIEVE_DATA"  
}
```

Se in qualsiasi momento quei nodi di riferimento sono completamente e correttamente caricati, l'azione di caricamento viene attivata

Questo, naturalmente, può attivare le azioni di caricamento di altri nodi

Controllo costante su ogni azione della tua interfaccia

Ogni nodo mantiene un registro di ciò che gli è successo.

```
"eventsLog": [  
  {  
    "what": "INSTANTIATED",  
    "when": "2024-06-26T14:01:41.864Z"  
  },  
  {  
    "what": "RECEIVED undefined FROM region",  
    "when": "2024-06-26T14:01:41.885Z"  
  },  
  {  
    "what": "RECEIVED undefined FROM province",  
    "when": "2024-06-26T14:01:41.886Z"  
  },  
  {  
    "what": "RECEIVED null FROM region",  
    "when": "2024-06-26T14:01:41.895Z"  
  },  
  {  
    "what": "RECEIVED Toscana FROM region",  
    "when": "2024-06-26T14:01:43.898Z"  
  },  
  {  
    "what": "RECEIVED null FROM province",  
    "when": "2024-06-26T14:01:43.900Z"  
  },  
  {  
    "what": "RECEIVED Prato FROM province",  
    "when": "2024-06-26T14:01:44.893Z"  
  },  
  {  
    "what": "GET REQ SENT: http://localhost:3000/retrieve/Toscana/Prato/city/list",  
    "when": "2024-06-26T14:01:44.894Z"  
  },  
  {  
    "what": "SELECTION INVALIDATED, PROPAGATED NULL",  
    "when": "2024-06-26T14:01:44.895Z"  
  },  
  {  
    "what": "CHOICES SET TO: Poggio a Caiano, Montemurlo, Carmignano",  
    "when": "2024-06-26T14:01:44.895Z"  
  }  
]
```

Aggiornamenti da altri nodi impostati come sue dipendenze

Azione di caricamento eseguita quando le dipendenze sono soddisfatte



wavebinder™

Grazie