

Wybe

**Making formal  
verification  
accessible to all**

- What is formal verification?

```
2 + 2 = 4
```

```
if ¬true then A else B  
= { applying ¬ }  
if false then A else B  
= { semantics of the if statement }  
B
```

- How can it be done






```
1 +1-length :  $\forall \{ \ell \} \{ A : \text{Set } \ell \} (h : A) (t : \mathbb{L} A) \rightarrow \text{length } (h :: t) \equiv \text{length } t + 1$   
2 +1-length h [] = refl  
3 +1-length h t = refl
```

- Wybe, an accessible and powerful tool

```
1 proof {  
2   theorem "double negation" (!(!x) == x)  
3   !(!x) == x  
4   ``==`` { ``GS 3.11`` }  
5   !x == !x  
6   ``==`` { ``== ident`` }  
7   True  
8 }
```

# Post interactions X

**Luis**   
@lamg\_dev


Promote  



[#fsharp](#) now we have a proof checker embedded in F#'s computation expressions WIP

[Traducir con DeepL](#) 

```
1 let ``double negation`` =  
2   proof () {  
3     Theorem("double negation", !(!x) == x)  
4     WithLaws [ trueTheorem ]  
5     !(!x) == x  
6     ``==`` { ``GS 3.11`` }  
7     !x == !x  
8     ``==`` { ``== ident`` }  
9     True  
10  }
```

1:36 PM · Apr 6, 2025 · **3,092** Views


 View post engagements


 1  13  73  9 


# Post interactions in LinkedIn


## Reactions


All 25


 19


 2


 2


 2


**Guillaume Claret** · 1st  
Security Researcher @Formal Land - Formal verification for Web3 - Solidity, Rust, OCaml

**Jonas Lara** · 1st  
Full Stack Developer

**Vinícius Gajo Marques Oliveira** · 1st  
Software & Infrastructure Engineer | Open-source contributor | DevOps/DevSecOps | Mechatronics Engineer

**Don Syme** · 1st  
Principal Researcher, Visiting Professor

**John A.** · 2nd  
Generative AI · Quantum Computing · Functional Programming · Cloud Computing · Scientific Computing · Principal Software Engineer @ Microsoft

























**Marius Fersigan** · 2nd

6

# Stars in GitHub

All 28

You know 9

 <b>muqluhan</b> ClinWise Follow	 <b>GuuD</b> Kyiv, Ukraine Follow	 <b>lamg</b> Germany Follow
 <b>dv-ar</b> Joined on Nov 24, 2022 Follow	 <b>ma3yta</b> Lviv, Ukraine Unfollow	 <b>stanislav</b> Berlin, Germany Unfollow
 <b>ibrahim324</b> sked Software GmbH Follow	 <b>Thorium</b> London, UK Unfollow	 <b>simontreanor</b> Finlar Follow
 <b>bdaniel7</b> Bucharest, Romania Follow	 <b>Reenuay</b> Joined on Aug 16, 2016 Follow	 <b>esneko</b> Latvia Follow
 <b>jasonmc</b> Brooklyn, New York Follow	 <b>jhwohlgemuth</b> Oak Ridge National Laboratory Follow	 <b>sgoguen</b> Joined on Mar 5, 2009 Follow
 <b>Jonas1ara</b> Mexico Unfollow	 <b>alsnrbernardo</b> Joined on May 26, 2020 Follow	 <b>clarus</b> Formal Land Unfollow
 <b>bretanac93</b> @sumup Unfollow	 <b>sheepia</b> Japan Follow	 <b>Numpy</b> UK Follow
 <b>shubhamkumar13</b> India Follow	 <b>NatElkins</b> Nelknet Follow	 <b>dawedawe</b> Freelance Unfollow

# The team

- 10 years ago

"Llegué a este estilo por mi preocupación de escribir los programas bien, pues los errores se me escapaban al escribir un programa y notaba que estaban ahí cuando los ejecutaba", afirma.



*Luis Ángel Méndez, un bicho raro entre sus colegas. Foto: Eduardo González Martínez*

El método Dijkstra se opone a la práctica imperante de escribir un programa primero y someterlo después a una corrección matemática. Con su propuesta, aunque más demorada, se obtiene un programa listo para ejecutarse, correcto desde su construcción.