Proseminar on computer-assisted mathematics

Session 7 - Introduction to Lean

LEAN
THEOREM PROVER

Florent Schaffhauser
Heidelberg University, Summer semester 2023

Today we will:

- Learn about Lean Theorem Prover, a programming language and proof assistant created by Leonardo de Moura in 2013.
- Get acquainted with Lean 3 and prove our first propositions, which will all be equalities between objects.

Resources about Lean

We will be using Lean 3. The best place to get started is the Lean community website:

https://leanprover-community.github.io/

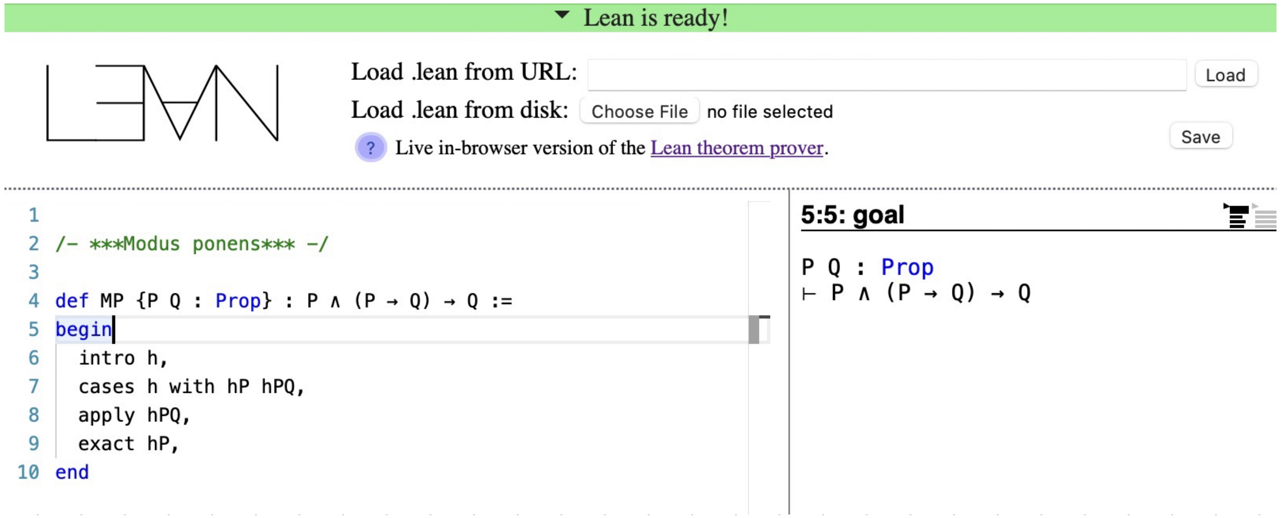
In particular the installation instructions.

https://leanprover-community.github.io/get_started.html

Other ways to use Lean (no installation)

In a web browser:

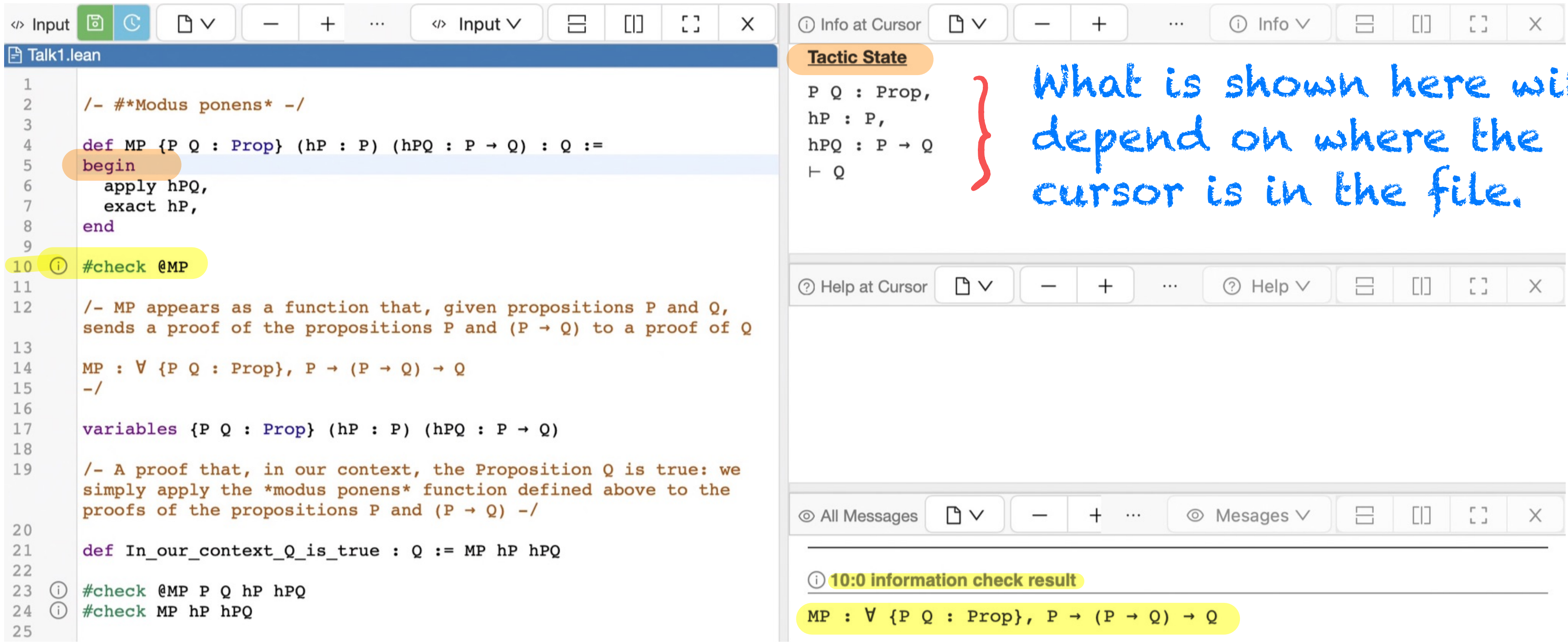
https://leanprover-community.github.io/lean-web-editor



The Lean web editor interface

Usage: move the cursor in the Lean file and read the response from the program on the right.

For us, the most convenient option today will be to upload our Lean file to CoCalc:



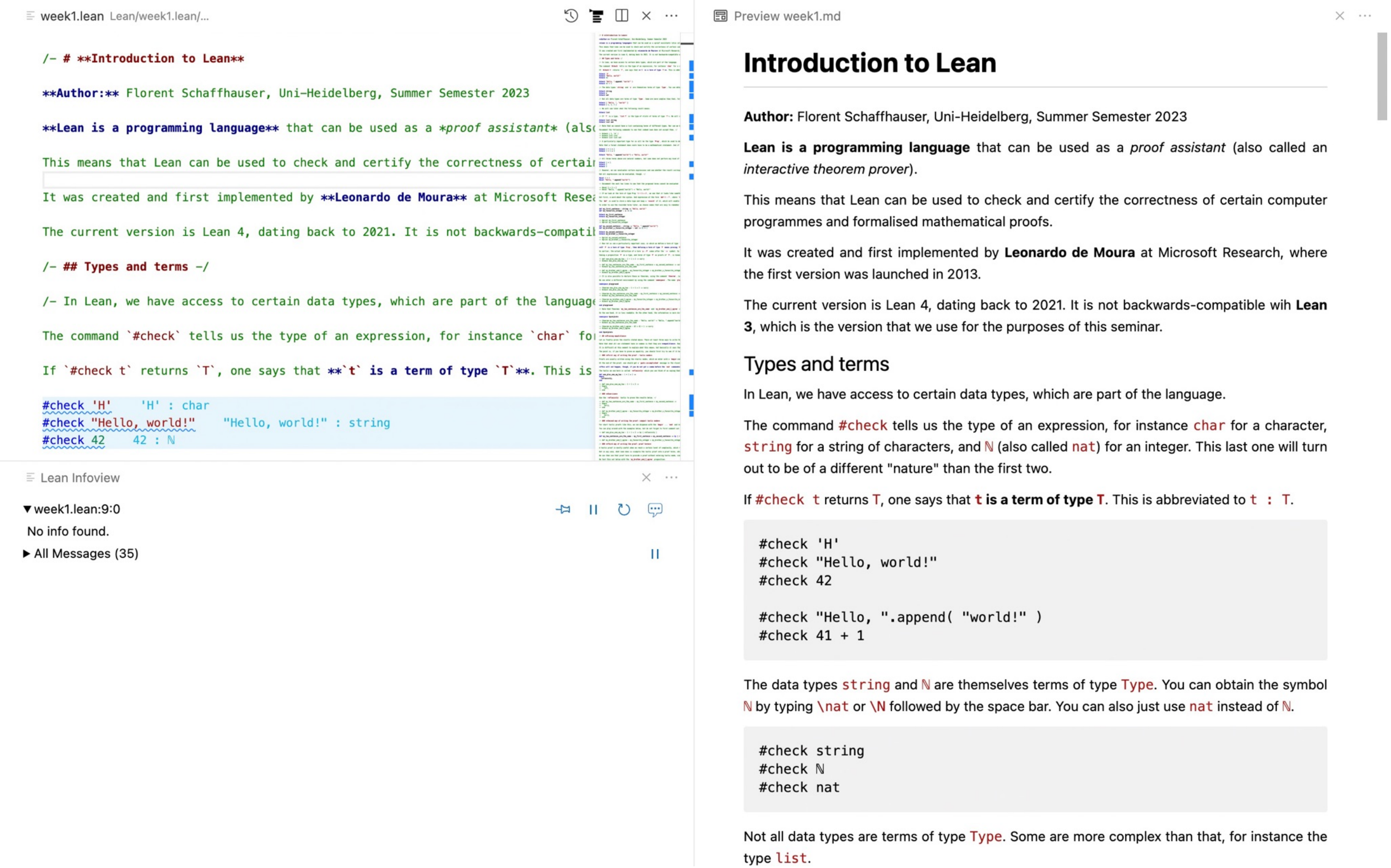
What is shown here will depend on where the cursor is in the file.

The Lean file

The data stored in the object called MP, defined between Lines 4 and 8.

Now we practice!

The first practice file is the Introduction to Lean file, available from the seminar webpage.



Introduction to Lean

Author: Florent Schaffhauser, Uni-Heidelberg, Summer Semester 2023

Lean is a programming language that can be used as a *proof assistant* (also called an *interactive theorem prover*).

This means that Lean can be used to check and certify the correctness of certain computer programmes and formalised mathematical proofs.

It was created and first implemented by **Leonardo de Moura** at Microsoft Research, where the first version was launched in 2013.

The current version is Lean 4, dating back to 2021. It is not backwards-compatible wih **Lean 3**, which is the version that we use for the purposes of this seminar.

Types and terms

In Lean, we have access to certain data types, which are part of the language.

The command `#check` tells us the type of an expression, for instance `char` for a character, `string` for a string of characters, and `ℕ` (also called `nat`) for an integer. This last one will turn out to be of a different "nature" than the first two.

If `#check t` returns `T`, one says that **`t` is a term of type `T`**. This is abbreviated to `t : T`.

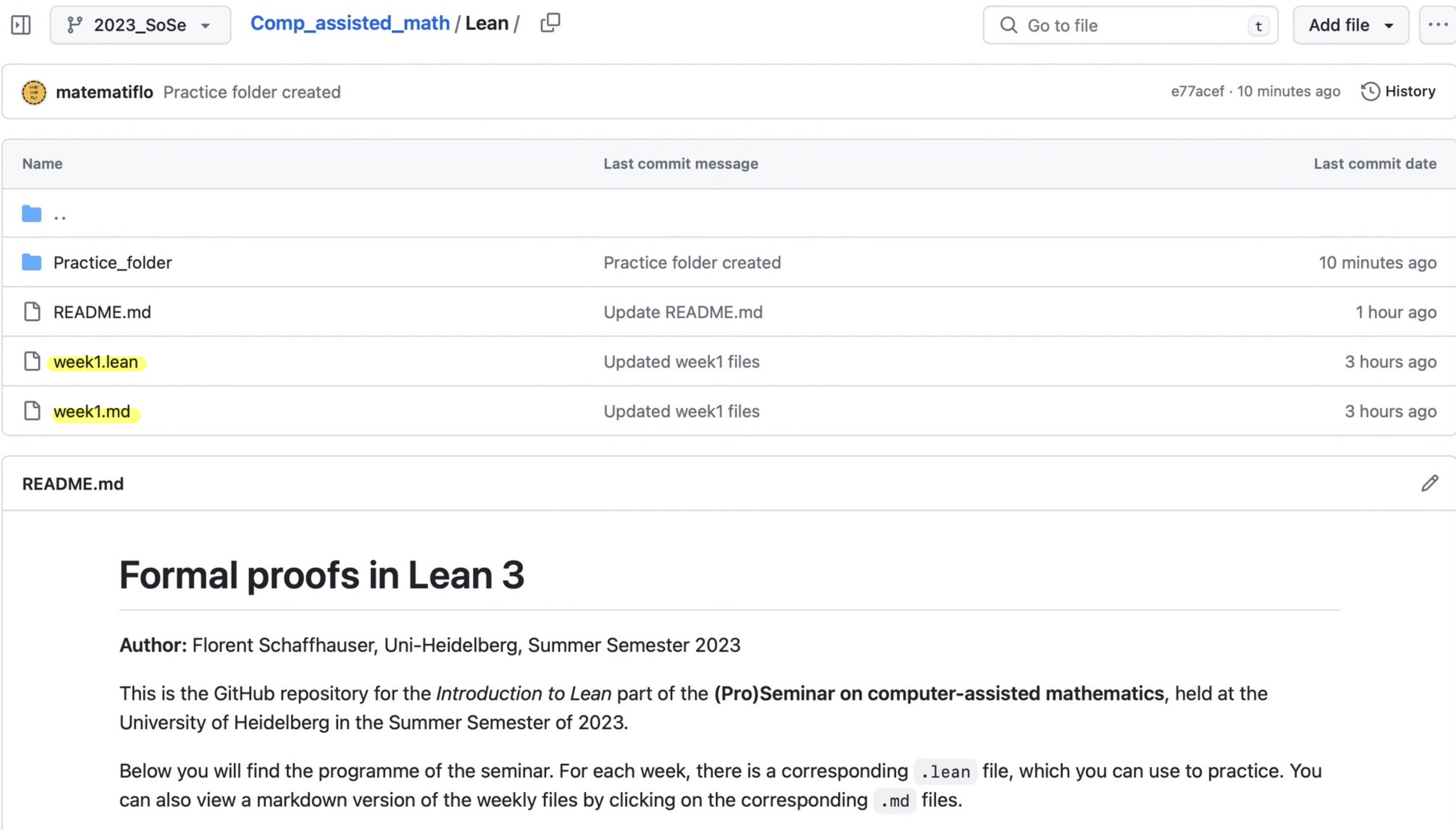
```
#check 'H'
#check "Hello, world!"
#check 42

#check "Hello, ".append( "world!" )
#check 41 + 1
```

The data types `string` and `ℕ` are themselves terms of type `Type`. You can obtain the symbol `ℕ` by typing `\nat` or `\N` followed by the space bar. You can also just use `nat` instead of `ℕ`.

```
#check string
#check ℕ
#check nat
```

Not all data types are terms of type `Type`. Some are more complex than that, for instance the type `list`.

On the GitHub repository of the seminar, you can find that file under the name week1.lean, along with its markdown version.



2023_SoSe Comp_assisted_math / Lean /

Go to file t | Add file

matematiflo Practice folder created e77acef · 10 minutes ago History

Name	Last commit message	Last commit date
..		
Practice_folder	Practice folder created	10 minutes ago
README.md	Update README.md	1 hour ago
week1.lean	Updated week1 files	3 hours ago
week1.md	Updated week1 files	3 hours ago

README.md

Formal proofs in Lean 3

Author: Florent Schaffhauser, Uni-Heidelberg, Summer Semester 2023

This is the GitHub repository for the *Introduction to Lean* part of the **(Pro)Seminar on computer-assisted mathematics**, held at the University of Heidelberg in the Summer Semester of 2023.

Below you will find the programme of the seminar. For each week, there is a corresponding `.lean` file, which you can use to practice. You can also view a markdown version of the weekly files by clicking on the corresponding `.md` files.

I have also put the modus ponens file in the Practice folder.