Frequently asked questions about course 1

Q: Is there a difference between `Prop` and `bool`?

A: Yes, there is a strong difference between the type `Prop` and the type `bool` in Coq whose logic is intuitionistic.

The type `bool` is computational. One can perform case analysis on its values and define functions on it. On the other side, one cannot universally quantify over it when the quantification domain is infinite since such a quantification would require infinite time and hence be non computable.

The type `Prop` of propositions is not computational. Propositions support universal quantification, even when the quantification domain is infinite, but propositions are not decidable in the sense that one cannot observe whether a proposition is true or false.

The type `bool` easily inject to `Prop` using the map

```
Definition bool_to_Prop (b:bool) : Prop := (b = true).
```

The converse would be possible by assuming classical logic, i.e. excluded-middle `EM : forall A , A \/ ~A` (requiring the library `Classical`). Then, indeed, we can define

```
Definition Prop_to_bool (A:Prop) : bool :=
  match EM A with
```

```
  | or_introl _ => true
  | or_intror _ => false
  end.
```

But then, we loose computation in Coq because `EM` has here no computational content.

Q: Is `bool -> True` a proposition?

A: In Coq, this is indeed a valid proposition and it is just an alternative notation for the proposition `forall b:bool, True`, since in general `A -> B` is just a notation for `forall a:A, B` when `x` does not occur in `B`, as witnessed by issuing the command

```
Check forall b:bool, True.
```

which outputs `bool -> True`.

Q: Is `True -> True` really the same as `forall a:True , True` in Coq?

A: Yes, both expressions coincides in Coq.

Q: Why to use `intro` to prove an implication?

A: Coq's logic is based on so-called natural deduction, which was designed by Gentzen in the 1930's. In natural deduction, proving an implication exactly amounts to assuming the premise of the implication and prove the conclusion. This is exactly what `intro` does: it assumes the premise (putting it "above the goal line") and leaves the conclusion as remaining goal.

Q: Why to use `intro` to prove a universal quantification?

A: Similarly, in natural deduction, proving a universal quantification amounts to taking some arbitrary variable in the domain of quantification and to prove the body of the universally quantified proposition under the assumption of existence of such an arbitrary inhabitant in the domain of quantification.

Q: The default navigation bindings in CoqIDE are intercepted by the window manager and do not work. What can I do?

A: On Linux, a suggestion is to change them to Alt-Shift in the Edit menu, Preferences item, Shortcut tab. This is done by clicking on Ctrl to deactivate it in the Navigation line, and clicking on Alt to activate this modifier. You will have however to save buffers and quit, then restart CoqIDE.