

# Coq 入門

# 定理証明支援系とは

- 証明が正しいことを検証してくれる装置。
- 公理から定理の証明を構築する手順を、型システムで模倣する。
- (証明は人間が作る)
- 検証だけでなく、証明の構築も支援してくれる。

定理証明支援系の1つとして **Coq** がある。

# インストール

Coq 8.8.0 まではインストーラが用意されている。最新は 8.16.0

- 新しさより簡単さ重視する人：[レポジトリ](#)からインストーラを入手
- 最新版を入りたい人：[Coq/SSReflect/MathCompの設定](#)に従う
  - `sudo apt get install`でなく `sudo apt install`
  - WSLでは「新しいライブラリの設定が要るかも」の箇所で `sudo apt install -y libgmp-dev` が必要だった([参考](#))

# エディタ

- **VScode+VSCoq** がおすすめ。VScode上で拡張機能(`Ctrl+Shift+X`)から「vscoq」と検索すれば出てくる。
  - Windowsの人はWSL上で動かせるように **Remote - WSL** も入れよう（「wsl」で検索）。
- **CoqIDE** もインストーラから入れた場合は利用できる（がVSCoqの方が便利）
- **Emacs** 上で **Proof General** を動かす手段もあるらしい。
- **jsCoq** : ブラウザ上で動く Coq。

# まずは簡単な例。

```
Section ModusPonens.  
Variables P Q : Prop.  
Theorem MP : P -> (P -> Q) -> Q.  
Proof.  
  intros Pis PimpliesQ.  
  apply PimpliesQ.  
  apply Pis.  
Qed.
```

VSCoq, jsCoq上では **Alt+↓** で1文ずつ証明が読み込まれる。

ここで `Print MP.` を実行すると

```
MP =  
fun (Pis : P) (PimpliesQ : P -> Q) =>  
PimpliesQ Pis  
  : P -> (P -> Q) -> Q  
  
Arguments MP _ _%function_scope
```

のように表示される。

つまり `MP` とは、引数として `Pis` (値) と `PimpliesQ` (関数) を受け取って `PimpliesQ Pis` を返す関数。

Coqの(正確には**Gallina**の)書式は `変数名 : 型`。

`MP` の型 `P -> (P -> Q) -> Q` は、`P` 型の値と `P->Q` 型の値(関数)を受け取って `Q` 型の値を返す関数、を意味する。

これを、「`P`という言明の証明と`P->Q`という言明の証明を受け取って`Q`という言明の証明を作る」ことと同一視する。

→ **Curry-Howard同型対応**

# リンク集

- [Coq クイックリファレンス](#)  
事例集をみればとりあえず証明が書ける。
- [TopProver](#)  
競プロのCoq版。最近はコンテストは開かれていないが過去問で証明の練習ができる。
- [Coq/SSReflect/MathComp Tutorial](#)  
スライドとかExample がいろいろ載ってる。
- [ソフトウェアの基礎](#)  
網羅的に勉強できそう。



# 参考文献

- [萩原学/アフェルト・レナルド『Coq/SSReflect/MathCompによる定理証明』\(森北出版, 2018\). 必携。](#)
- [定理証明支援系 Coq による形式検証 \(PDF\)](#)