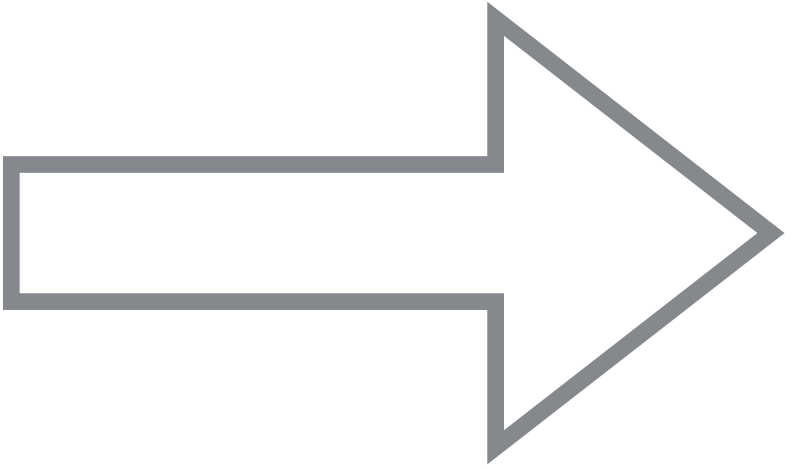
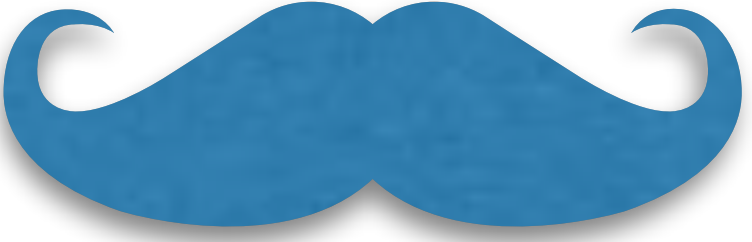
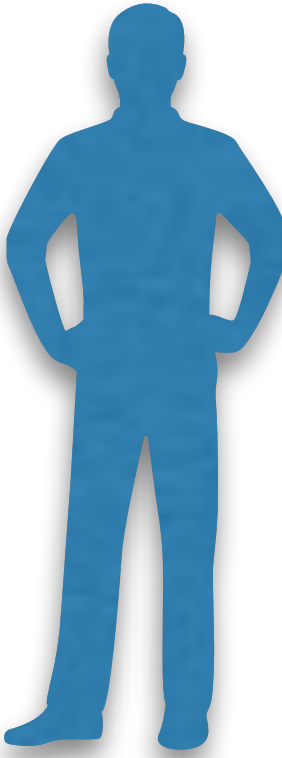
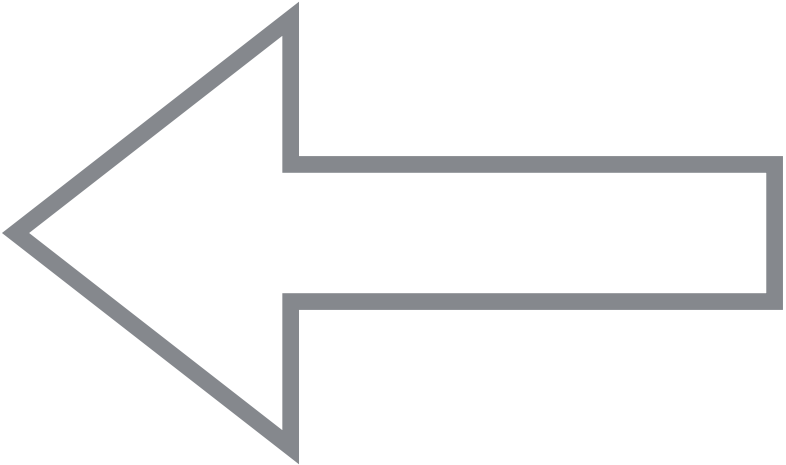
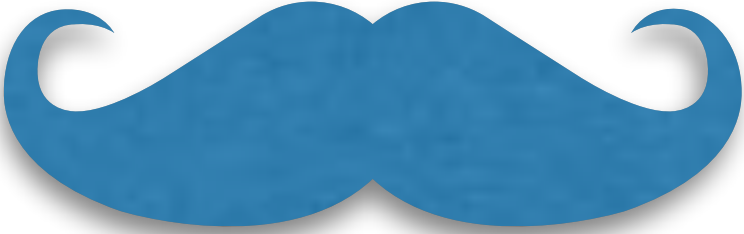


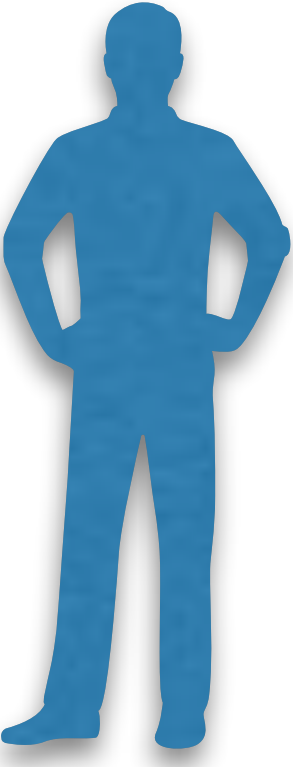




- Adding new functions may break existing code (name clashes)
- Even a “simple” rename is far from trivial:
  - Prone to silently change code behaviour (C++, D, …)
  - Only applies in one branch, causing evident and undetected merge conflicts





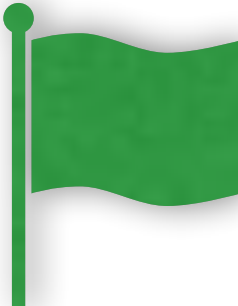


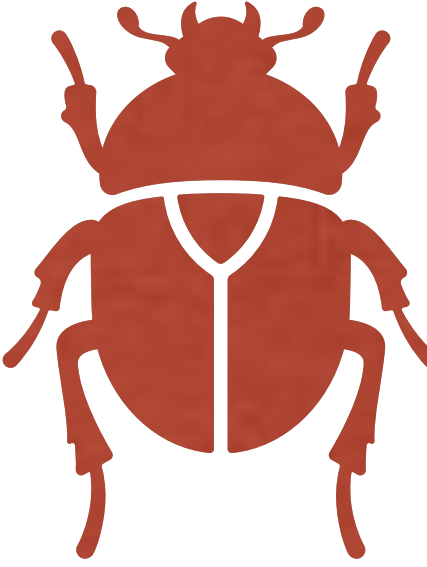
- Adding new functions never breaks existing code
- Clashes are a presentation issue - how to display them?
- Renames are simple and reliable
- Enables localisation - Identifiers may have names in different languages (English, French, etc) enabling diverse collaboration

Name Resolution:

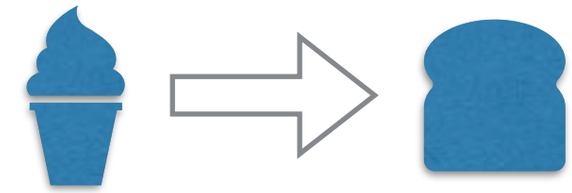
Name Presentation:

?



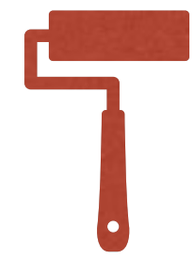


Syntactic Sugar - Today

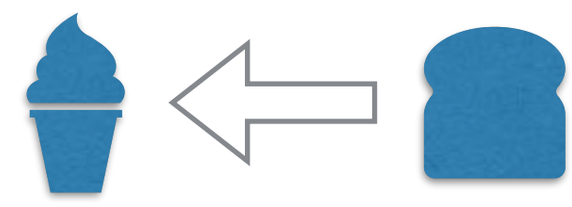


Syntactic sugars allow the programmer to write *sweeter* code. In Scala `{_ * 2 + _}` is equivalent to `{(x,y) => x * 2 + y}`

These sugars apply in special patterns, small changes to the code require switching syntaxes



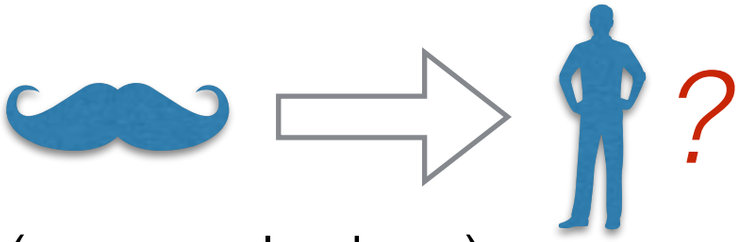
Projectional Syntactic Sugar



Code is presented sugared when it matches patterns

- Automatic reformatting of code
- Consistent style, less spurious choices

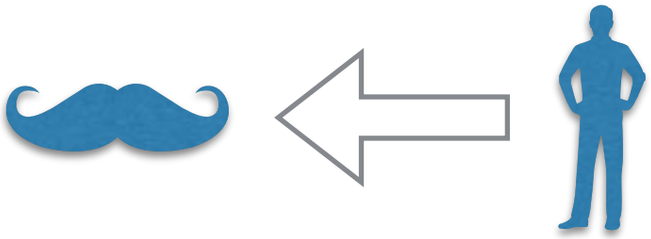
Name Resolution:



- Adding new functions may break existing code (name clashes)
- Even a “simple” rename is far from trivial:
  - Prone to silently change code behaviour (C++, D, …)

  - Only applies in one branch, causing evident and undetected merge conflicts

Name Presentation:



- Adding new functions never breaks existing code
- Clashes are a presentation issue - how to display them?
- Renames are simple and reliable
- Enables localisation - Identifiers may have names in different languages (English, French, etc) enabling diverse collaboration