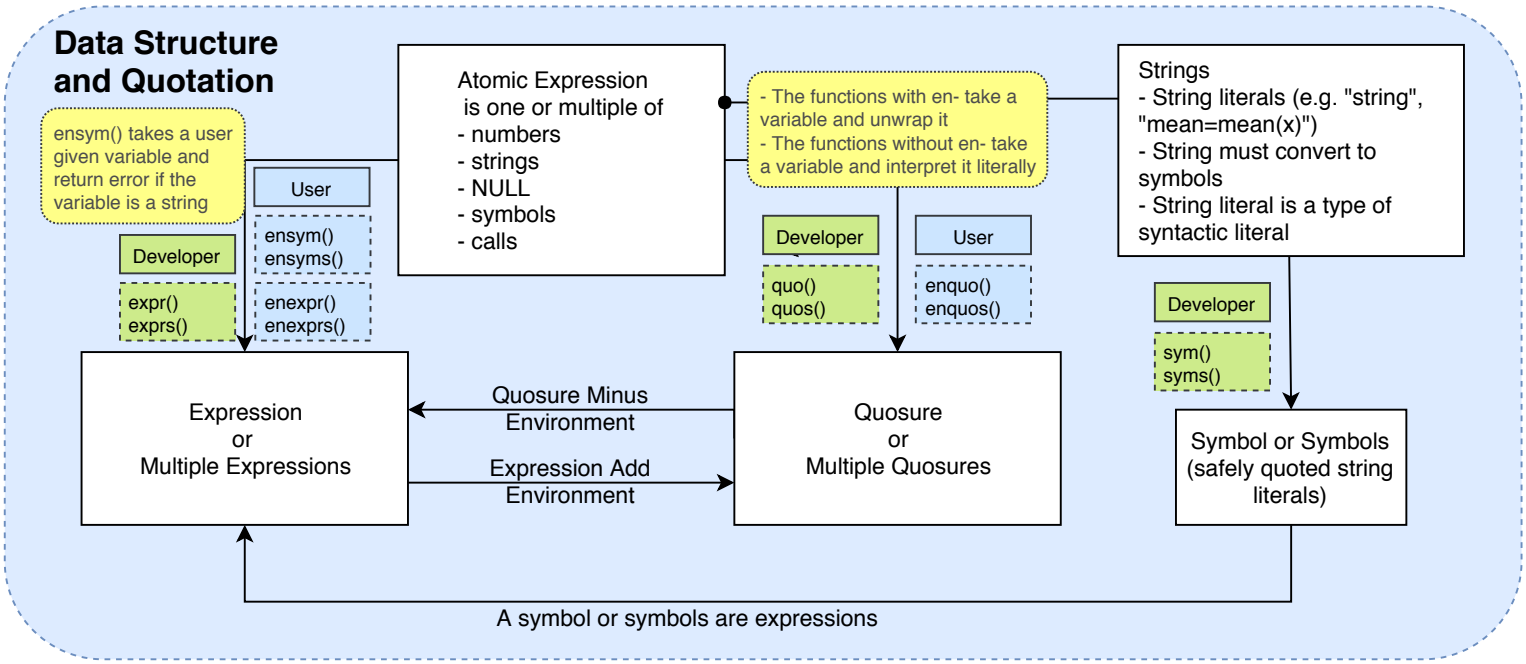
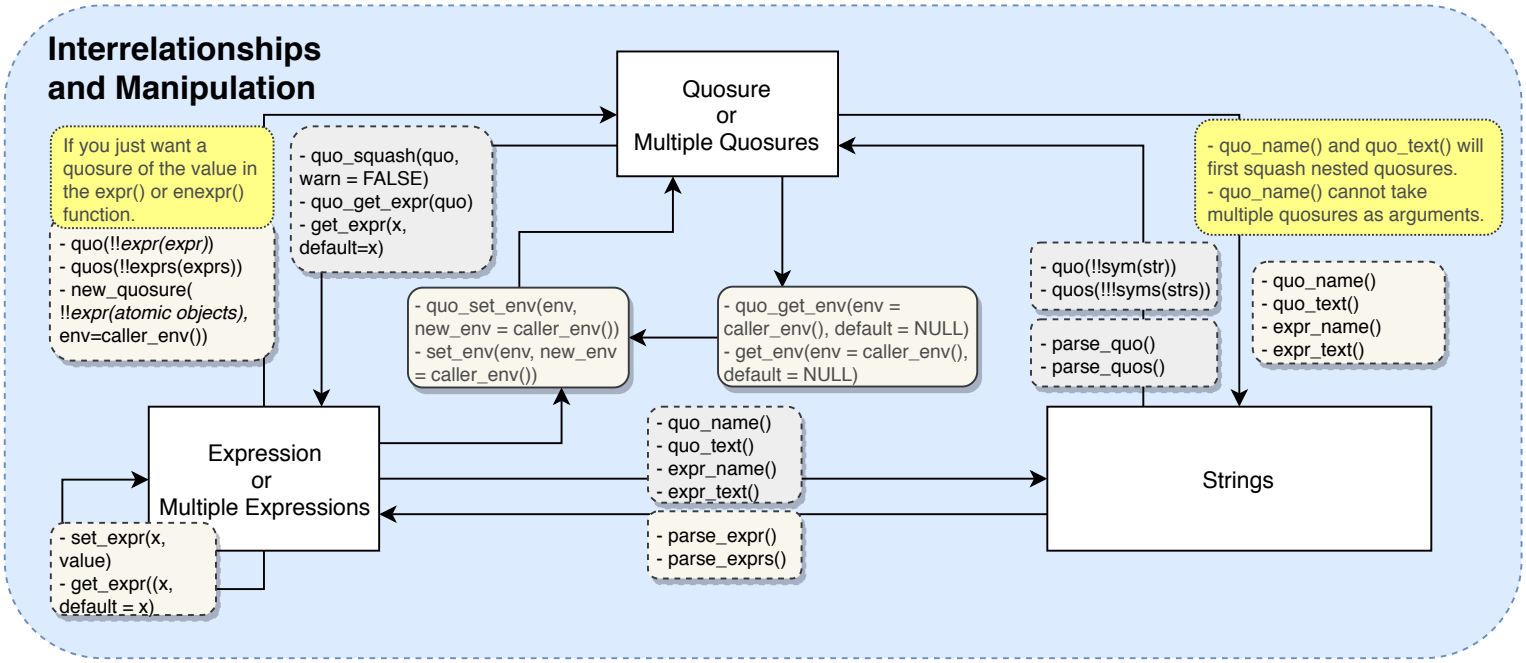
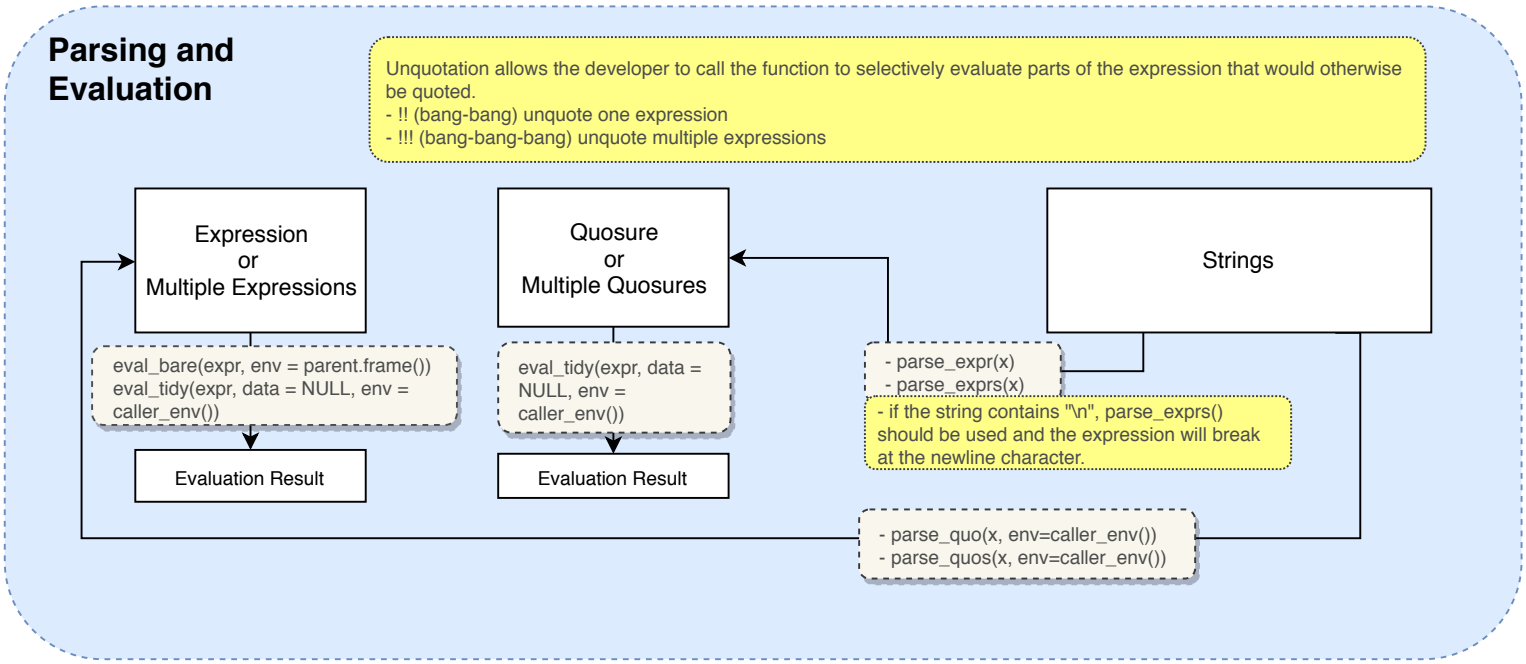

Data Structure and Quotation
ensym() takes a user given variable and return error if the variable is a string
Atomic Expression is one or multiple of
- numbers
- strings
- NULL
- symbols
- calls
- The functions with en- take a variable and unwrap it
- The functions without en- take a variable and interpret it literally
Strings
- String literals (e.g. "string", "mean=mean(x)")
- String must convert to symbols
- String literal is a type of syntactic literal
User
ensym()
ensyms()
Developer
expr()
exprs()
enexpr()
enexprs()
Developer
quo()
quos()
User
enquo()
enquos()
Developer
sym()
syms()
Expression or Multiple Expressions
Quosure Minus Environment
Expression Add Environment
Quosure or Multiple Quosures
Symbol or Symbols (safely quoted string literals)
A symbol or symbols are expressions


Interrelationships and Manipulation
Quosure or Multiple Quosures
If you just want a quosure of the value in the expr() or enexpr() function.
- quo(!!expr(expr))
- quos(!!exprs(exprs))
- new_quosure(!!expr(atomic objects), env=caller_env())
- quo_squash(quo, warn = FALSE)
- quo_get_expr(quo)
- get_expr(x, default=x)
- quo_name() and quo_text() will first squash nested quosures.
- quo_name() cannot take multiple quosures as arguments.
- quo(!!sym(str))
- quos(!!!syms(strs))
- quo_name()
- quo_text()
- expr_name()
- expr_text()
- quo_set_env(env, new_env = caller_env())
- set_env(env, new_env = caller_env())
- quo_get_env(env = caller_env(), default = NULL)
- get_env(env = caller_env(), default = NULL)
- parse_quo()
- parse_quos()
Expression or Multiple Expressions
- quo_name()
- quo_text()
- expr_name()
- expr_text()
Strings
- set_expr(x, value)
- get_expr((x, default = x)
- parse_expr()
- parse_exprs()


Parsing and Evaluation
Unquotation allows the developer to call the function to selectively evaluate parts of the expression that would otherwise be quoted.
- !! (bang-bang) unquote one expression
- !!! (bang-bang-bang) unquote multiple expressions
Expression or Multiple Expressions
Quosure or Multiple Quosures
Strings
eval_bare(expr, env = parent.frame())
eval_tidy(expr, data = NULL, env = caller_env())
eval_tidy(expr, data = NULL, env = caller_env())
- parse_expr(x)
- parse_exprs(x)
- if the string contains "\n", parse_exprs() should be used and the expression will break at the newline character.
Evaluation Result
Evaluation Result
- parse_quo(x, env=caller_env())
- parse_quos(x, env=caller_env())