Scoping Rules Examples

$[Char]$

Example 1 : Quantifier Scope

Variables introduced in quantifiers are local to that quantifier:

$\forall x : \mathbb{N} \bullet x > 0$
Here x is only in scope within the quantifier. Outside the quantifier, x is not defined.

Example 2 : Nested Quantifiers

$\forall x : \mathbb{N} \bullet \exists y : \mathbb{N} \bullet x + y = 10$
x is in scope for the entire outer quantifier, including the inner quantifier. y is only in scope within the $\exists$ quantifier.

Example 3 : Shadowing elem Nested Quantifiers

$\forall x : \mathbb{N} \bullet x > 0 \Rightarrow (\exists x : \mathbb{N} \bullet x < 10)$
The inner x shadows the outer x. Within the $\exists$ quantifier, x refers to the inner binding. This is allowed but can be confusing.

Example 4 : Set Comprehension Scope

$\{ x : \mathbb{N} \mid x > 0 \land x < 10 \}$
x is in scope only within the set comprehension. The constraint $x > 0$ and $x < 10$ uses the same x.

Example 5 : Bullet Separator Scope

$\{ x : \mathbb{N} \mid x < 5 \bullet x * x \}$
x is in scope both before and after the bullet. The constraint $x < 5$ and the term x * x refer to the same x.

Example 6 : Schema Component Scope

Counter

$count : \mathbb{N}$
$limit : \mathbb{N}$

$count \leq limit$

Within the schema, count and limit are in scope in the where clause. Outside the schema, they must be accessed via schema references.

Example 7 : Schema Reference

$$c : Counter$$

$$c.count = 5$$
$$c.limit = 10$$

When referencing schema components from outside, use dot *notation* : $c.count$ and c.limit.

Example 8 : Multiple Schemas - No Name Conflict

Account

$$balance : \mathbb{N}$$

$$balance \geq 0$$

Transaction

$$balance : \mathbb{N}$$

$$balance \geq 0$$

Both schemas have a component named balance. This is allowed because they are in different schema scopes.

Example 9 : Schema Inclusion Scope

User

$$userId : \mathbb{N}$$
$$name : \text{seq } Char$$

LoggedInUser

$$userId : \mathbb{N}$$
$$userName : \text{seq } Char$$
$$sessionToken : \mathbb{N}$$

$$sessionToken > 0$$

LoggedInUser has its own copies of userId and name fields (as userId and userName), plus sessionToken. In full Z notation, schema inclusion would allow referencing User directly, but here we show explicit declarations.

Example 10 : Global vs Local Names

$[GlobalType]$

$$globalVar : \mathbb{N}$$

$$globalVar = 42$$

LocalScope

$$x : \mathbb{N}$$

$$x = globalVar + 1$$

globalVar is in the global scope, accessible from within schemas. x is local to LocalScope.

Example 11 : Quantifier with Global Names

$$maximum : \mathbb{N}$$
$$maximum = 100$$

Bounded
$$value : \mathbb{N}$$
$$\forall x : \mathbb{N} \bullet x \leq maximum \Rightarrow value \leq x$$

The quantifier introduces local x, but maximum refers to the global axdef constant.

Example 12 : Name Capture Avoidance

Be careful with name shadowing in complex expressions:

$$x : \mathbb{N}$$
$$x = 10$$
$$\forall x : \mathbb{N} \bullet x > 0 \Rightarrow x \geq 0$$

The global $x = 10$, but the quantifier in the where clause uses a local x that shadows the global x within its scope.

Example 13 : Lambda Scope

$\lambda x : \mathbb{N} \bullet x * x$
x is in scope only within the lambda body (x * x). The lambda introduces its own scope for the parameter.

Example 14 : Nested Lambdas

$\lambda x : \mathbb{N} \bullet \lambda y : \mathbb{N} \bullet x + y$
x is in scope in both the outer lambda body and the inner lambda body. y is only in scope in the inner lambda body.

Example 15 : Mu Operator Scope

$(\mu x : \mathbb{N} \mid x * x = 25)$
x is in scope only within the mu expression (the constraint and any expression after a bullet).

Example 16 : Set Comprehension with Multiple Variables

$\{ x, y : \mathbb{N} \mid x < y \land y < 10 \bullet (x, y) \}$
Both x and y are in scope throughout the set comprehension, including the constraint and the term.

Example 17 : Schema Quantification

Database
$records : \mathbb{P}\,\mathbb{N}$

$\exists\, Database \bullet \#records > 100$
This quantifies over schema Database. The components of Database (records) are in scope within the predicate.

Example 18 : Scope Hygiene Best Practices

To avoid confusion:

1. Don't shadow names unnecessarily—use different variable names

2. Keep quantifier nesting shallow when possible

3. Use descriptive names that make scope obvious

4. Avoid reusing schema component names in nested contexts

5. Document intentional shadowing with comments

Example 19 : Good Style (Distinct Names)

$\forall\, x : \mathbb{N} \bullet x > 0 \Rightarrow (\exists\, y : \mathbb{N} \bullet y < x \land y > 0)$
Uses distinct names x and y—clear and unambiguous.

Example 20 : Poor Style (Shadowing)

$\forall\, x : \mathbb{N} \bullet x > 0 \Rightarrow (\exists\, x : \mathbb{N} \bullet x < 10)$
Inner x shadows outer x—confusing and error-prone. Avoid this pattern.

Example 21 : Scoping elem Definitions

Poor style example (not valid due to name shadowing):

If we had: axdef with square : $\mathbb{N} \longrightarrow \mathbb{N}$ and $n : N$, where $\forall\ n : N$ shadows the global n.

The quantifier's n would be local to the quantifier, and the global $n = 5$ would be separate.

This is legal but confusing. Better to use different names:

$square : \mathbb{N} \longrightarrow \mathbb{N}$
$testValue : \mathbb{N}$

$\forall\, x : \mathbb{N} \bullet square(x) = x * x$
$testValue = 5$

Now there's no shadowing—much clearer!

Example 22 : Schema Composition Scope

StateA

$x : \mathbb{N}$

StateB

$y : \mathbb{N}$

Combined

$x : \mathbb{N}$

$y : \mathbb{N}$

$x + y > 0$

Combined has fields x and y. In full Z notation, schema inclusion (StateA; StateB) would automatically bring in these fields from other schemas, but here we show explicit declarations.

Example 23 : Free Type Scope

$Color ::= red \mid green \mid blue$

ColoredItem

$color : Color$

$color \in \{red, blue\}$

The free type constructors (red, green, blue) are in global scope, accessible within schemas and definitions.