CSE 307: Principles of Programming Languages

Names, Scopes, and Bindings

R. Sekar

Topics

Bindings: Names and Attributes

- Names are a fundamental abstraction in languages to denote entities
- Meanings associated with these entities is captured via attributes associated with the names
- Attributes differ depending on the entity:
  - location (for variables)
  - value (for constants)
  - formal parameter types (functions)
- Binding: Establishing an association between name and an attribute.

Names

- **Names** or **Identifiers** denote various language *entities*:
  - Constants
  - Variables
  - Procedures and Functions
  - Types, ...
- Entities have *attributes*

Entity	*Example Attributes*
Constants	type, value, ...
Variables	type, location, ...
Functions	signature, implementation, ...

Attributes

- Attributes are associated with names (to be more precise, with the entities they denote).
- Attributes describe the *meaning* or *semantics* of names (and entities).

`int x;`	There is a variable, named `x`, of type integer.
`int y = 2;`	Variable named `x`, of type integer, with initial value 2.
`Set s = new Set();`	Variable named `s`, of type `Set` that refers to an object of class `Set`

- An *attribute* may be
  - *static:* can be determined at translation (compilation) time, or
  - *dynamic:* can be determined only at execution time.

Static and Dynamic Attributes

- `int x;`
  - The *type* of `x` can be statically determined;
  - The *value* of `x` is dynamically determined;
  - The *location* of `x` (the element in memory will be associated with `x`) can be statically determined if `x` is a global variable.
- `Set s = new Set();`
  - The *type* of `s` can be statically determined.
  - The *value* of `s`, i.e. the object that `s` refers to, is dynamically determined.

Static vs. Dynamic specifies the *earliest* time the attribute *can* be computed

. . . not when it *is* computed in any particular implementation.

Binding

"Binding" is the process of associating attributes with names.

- **Binding time** of an attribute: whether an attribute can be computed at translation time or only at execution time.
- A more refined classification of binding times:
  - **Static:**
    - Language definition time (e.g. `boolean`, `char`, etc.)
    - Language implementation time (e.g. `maxint`, `float`, etc.)
    - Translation time ("compile time") (e.g. value of `n` in `const int n = 5;`)
    - Link time (e.g. the definition of function `f` in `extern int f();`)
    - Load time (e.g. the location of a global variable, i.e., where it will be stored in memory)
  - **Dynamic:**
    - Execution time

Binding Time (Continued)

- Examples
  - type is statically bound in most langs
  - value of a variable is dynamically bound
  - location may be dynamically or statically bound
- Binding time also affects where bindings are stored
  - Name $\rightarrow$ type: symbol table
  - Name $\rightarrow$ location: environment
  - Location $\rightarrow$value: memory