Funcons-beta: Classes *

The PLanCompS Project

`Classes.cbs` | PLAIN | PRETTY

Classes

[*Datatype* classes
  Funcon class
  Funcon class-instantiator
  Funcon class-feature-map
  Funcon class-superclass-name-sequence
  Funcon class-name-tree
  Funcon is-subclass-name
  Funcon class-name-single-inheritance-feature-map]

Datatype classes ::= class(_ : thunks(references(objects)), _ : environments, _ : identifiers*)

class(*Thunk*, *Env*, *C**) is a class with: * a thunk *Thunk* for instantiating the class, * an environment *Env* with the features declared by the class, and * a sequence *C** of names of direct superclasses. class(*Thunk*, *Env*) is a base class, having no superclasses. class(*Thunk*, *Env*, *C*) is a class with a single superclass.

Class instantiation forces its thunk to compute a reference to an object.

Features are inherited from superclasses. When features with the same name are declared in simultaneously inherited classes, the order of the superclass identifiers in *C** may affect resolution of references to features. Overloading of feature names is supported by using type maps as features.

The class table is represented by binding class names to classes. The class superclass hierarchy is assumed to be acyclic.

Funcon class-instantiator(_ : classes) : ⇒ thunks(references(objects))
  Rule class-instantiator
    class(*Thunk* : thunks(_), *Envs* : environments, *C** : identifiers*) ⇝
    Thunk

Funcon class-feature-map(_ : classes) : ⇒ environments
  Rule class-feature-map
    class(*Thunk* : thunks(_), *Env* : environments, *C** : identifiers*) ⇝
    Env

---

*Suggestions for improvement: `plancomps@gmail.com`.
Reports of issues: `https://github.com/plancomps/CBS-beta/issues`.

Funcon class-superclass-name-sequence(_ : classes) : ⇒ identifiers*

Rule class-superclass-name-sequence
class(*Thunk* : thunks(_), *Env* : environments, C^* : identifiers*) ⇝
C^*

Funcon class-name-tree(_ : identifiers) : ⇒ trees(identifiers)

class-name-tree C forms a tree where the branches are the class name trees for the superclasses of C.

Rule class-name-tree(C : identifiers) ⇝
tree(
C,
interleave-map(
class-name-tree given,
class-superclass-name-sequence bound-value C))

Funcon is-subclass-name(C : identifiers, C' : identifiers) : ⇒ booleans
⇝ is-in-set(C, {forest-value-sequence class-name-tree C'})

The result of is-subclass-name(C, C') does not depend on the order of the names in forest-value-sequence class-name-tree C

Funcon class-name-single-inheritance-feature-map(C : identifiers) : ⇒ environments
⇝ map-override interleave-map(
class-feature-map bound-value given,
single-branching-sequence class-name-tree C)

For multiple inheritance, different resolution orders can be specified by using different linearisations of the class name tree.