Typestate
Type classes
Here are a couple examples.

purposes in programming languages. It's the basis of computation in

Unification, by the way, is a powerful technique, used for a variety of

There are many other such polymorphic functions.

Any time the ML or Haskell compiler determines that A and B

the same type, it tries to

helper

fib

something it can't do at the global level.)

the

the

6

5

3

complicated case: ML (or

or

Java uses

implementations (the ISO standard uses name equivalence).  Java uses

your language, the following might also be structurally equivalent to

Name equivalence is more fashionable these days, but not universal.

The

Type compatibility / type equivalence

Both forms of type equivalence have nontrivial implementation issues for

implementations (the ISO standard uses name equivalence).  Java uses

Name equivalence is based on declarations.  Structural

The

Type inference and polymorphism

Type state is not the only kind of polymorphism.

Structural typestate

is

type

collection of well

description down to the level

Typestate is a powerful concept that allows you to

Type inference and polymorphism:

Typestate

Type classes

Here are a couple examples.

purposes in programming languages. It's the basis of computation in

Unification, by the way, is a powerful technique, used for a variety of

There are many other such polymorphic functions.

Any time the ML or Haskell compiler determines that A and B

the same type, it tries to

helper

fib

something it can't do at the global level.)

the

the

6

5

3

complicated case: ML (or

or

Java uses

implementations (the ISO standard uses name equivalence).  Java uses

your language, the following might also be structurally equivalent to

Name equivalence is more fashionable these days, but not universal.

The

Type compatibility / type equivalence

Both forms of type equivalence have nontrivial implementation issues for

implementations (the ISO standard uses name equivalence).  Java uses

Name equivalence is based on declarations.  Structural

The

Type inference and polymorphism

Type state is not the only kind of polymorphism.

Structural typestate

is

type

collection of well

description down to the level

Typestate is a powerful concept that allows you to

Type inference and polymorphism:

Typestate

Type classes

Here are a couple examples.

purposes in programming languages. It's the basis of computation in

Unification, by the way, is a powerful technique, used for a variety of

There are many other such polymorphic functions.

Any time the ML or Haskell compiler determines that A and B

the same type, it tries to

helper

fib

something it can't do at the global level.)

the

the

6

5

3

complicated case: ML (or

or

Java uses

implementations (the ISO standard uses name equivalence).  Java uses

your language, the following might also be structurally equivalent to

Name equivalence is more fashionable these days, but not universal.

The

Type compatibility / type equivalence

Both forms of type equivalence have nontrivial implementation issues for

implementations (the ISO standard uses name equivalence).  Java uses

Name equivalence is based on declarations.  Structural

The

Type inference and polymorphism

Type state is not the only kind of polymorphism.

Structural typestate

is

type

collection of well

description down to the level

Typestate is a powerful concept that allows you to

Type inference and polymorphism: