Stabilize

and other integrity traits

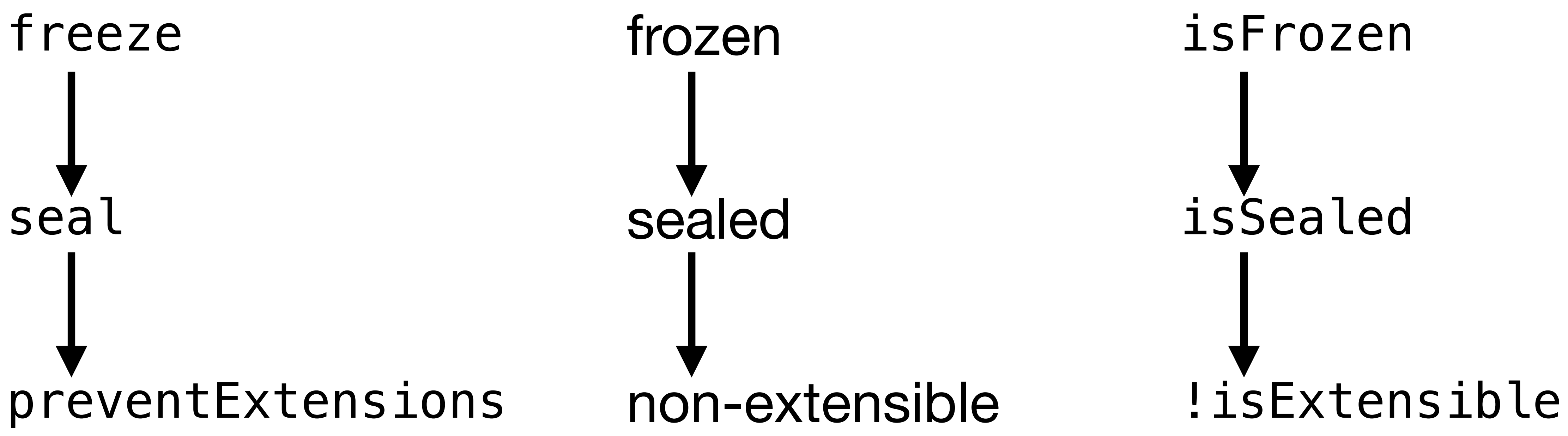
Mark S. Miller Agoric

105th Plenary

December 2024

TC 39

Current Integrity “Levels”



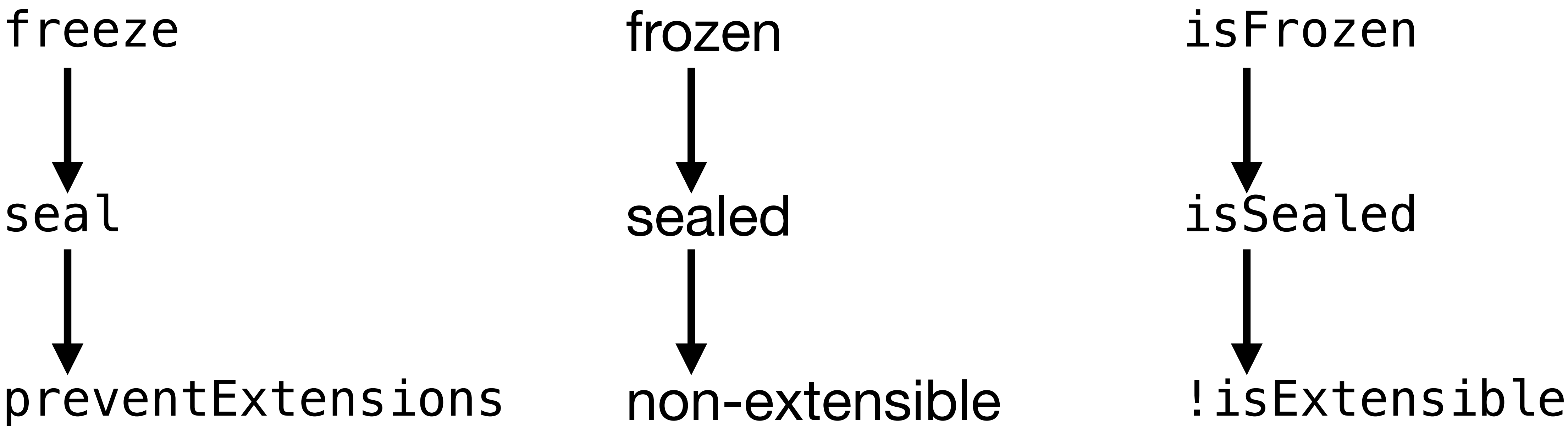
`freeze` → `seal` → `preventExtensions`

frozen → sealed → non-extensible

`isFrozen` → `isSealed` → `!isExtensible`

Current Integrity “Levels”

- Monotonic one-way switch



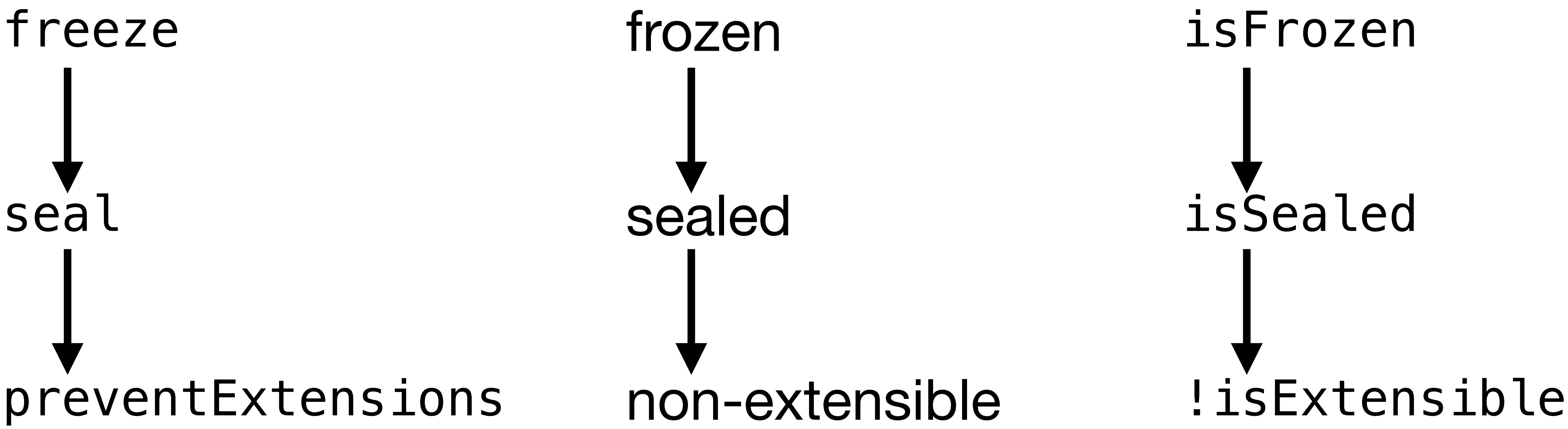
`freeze`
↓
`seal`
↓
`preventExtensions`

frozen
↓
sealed
↓
non-extensible

`isFrozen`
↓
`isSealed`
↓
`!isExtensible`

Current Integrity “Levels”

- Monotonic one-way switch
- Stronger object invariants



`freeze`

`seal`

`preventExtensions`

frozen

sealed

non-extensible

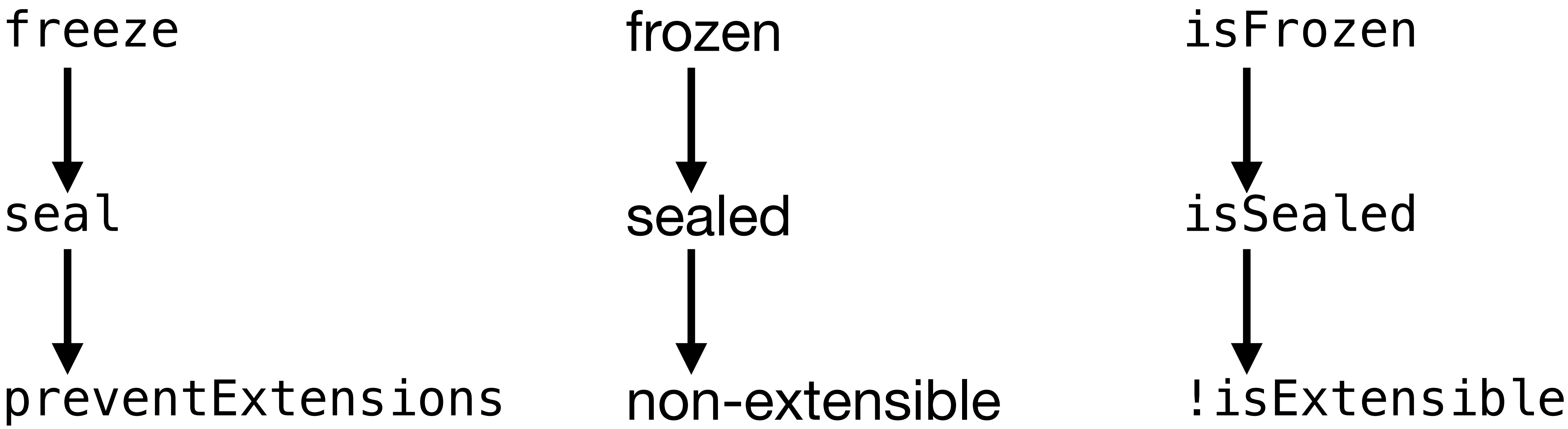
`isFrozen`

`isSealed`

`!isExtensible`

Current Integrity “Levels”

- Monotonic one-way switch
- Stronger object invariants
- Proxy is X iff target is X

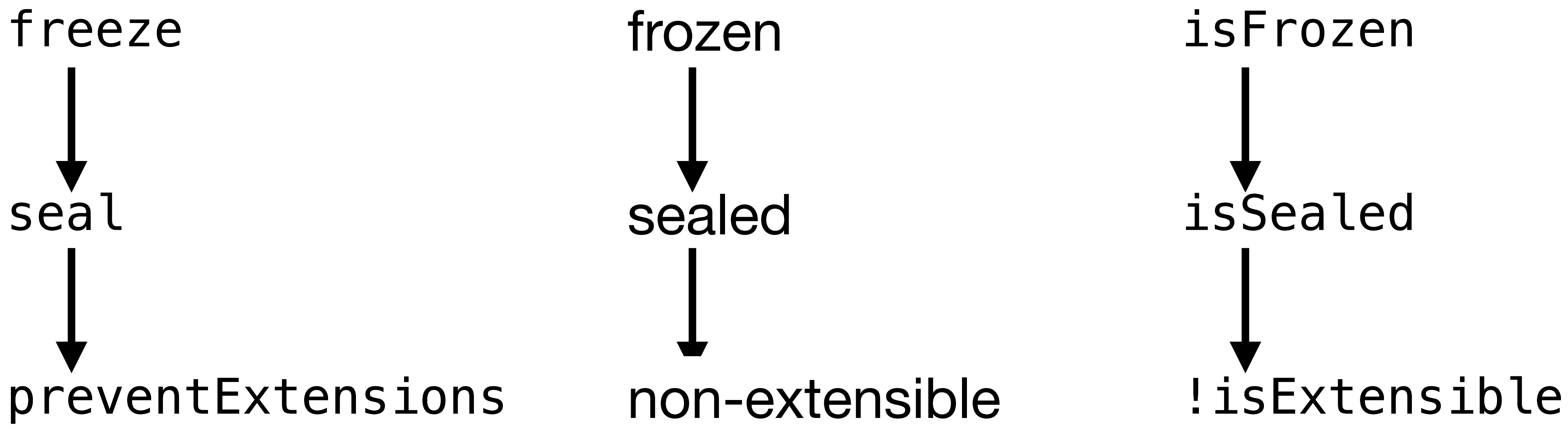


freeze → seal → preventExtensions

frozen → sealed → non-extensible

isFrozen → isSealed → !isExtensible

Current Integrity “Levels”



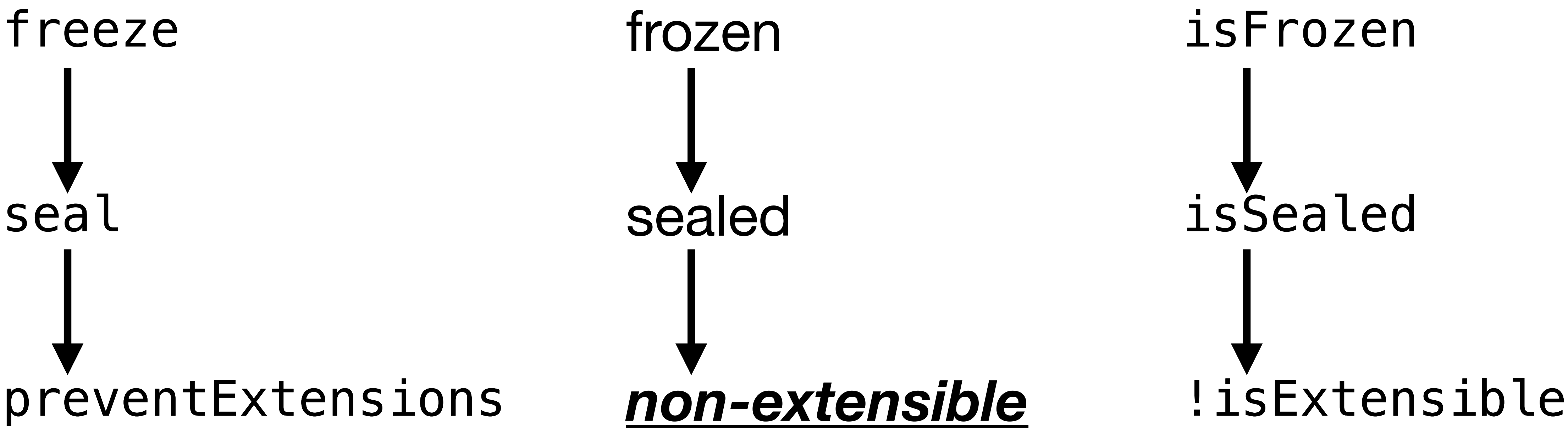
`freeze`
`seal`
`preventExtensions`

frozen
sealed
non-extensible

`isFrozen`
`isSealed`
`!isExtensible`

Current Integrity "Levels"

- ***Explicit*** vs Emergent



`freeze`

`seal`

`preventExtensions`

frozen

sealed

non-extensible

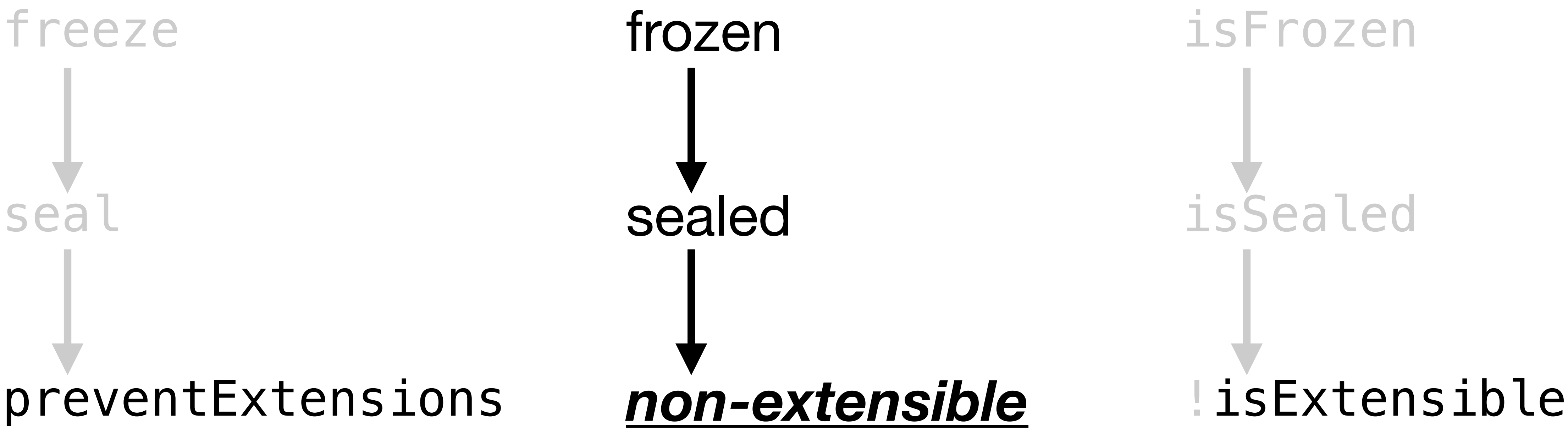
`isFrozen`

`isSealed`

`!isExtensible`

Current Integrity "Levels"

- ***Explicit*** vs Emergent
- 2 Proxy traps per ***explicit*** integrity "level"



`freeze`

`seal`

`preventExtensions`

frozen

sealed

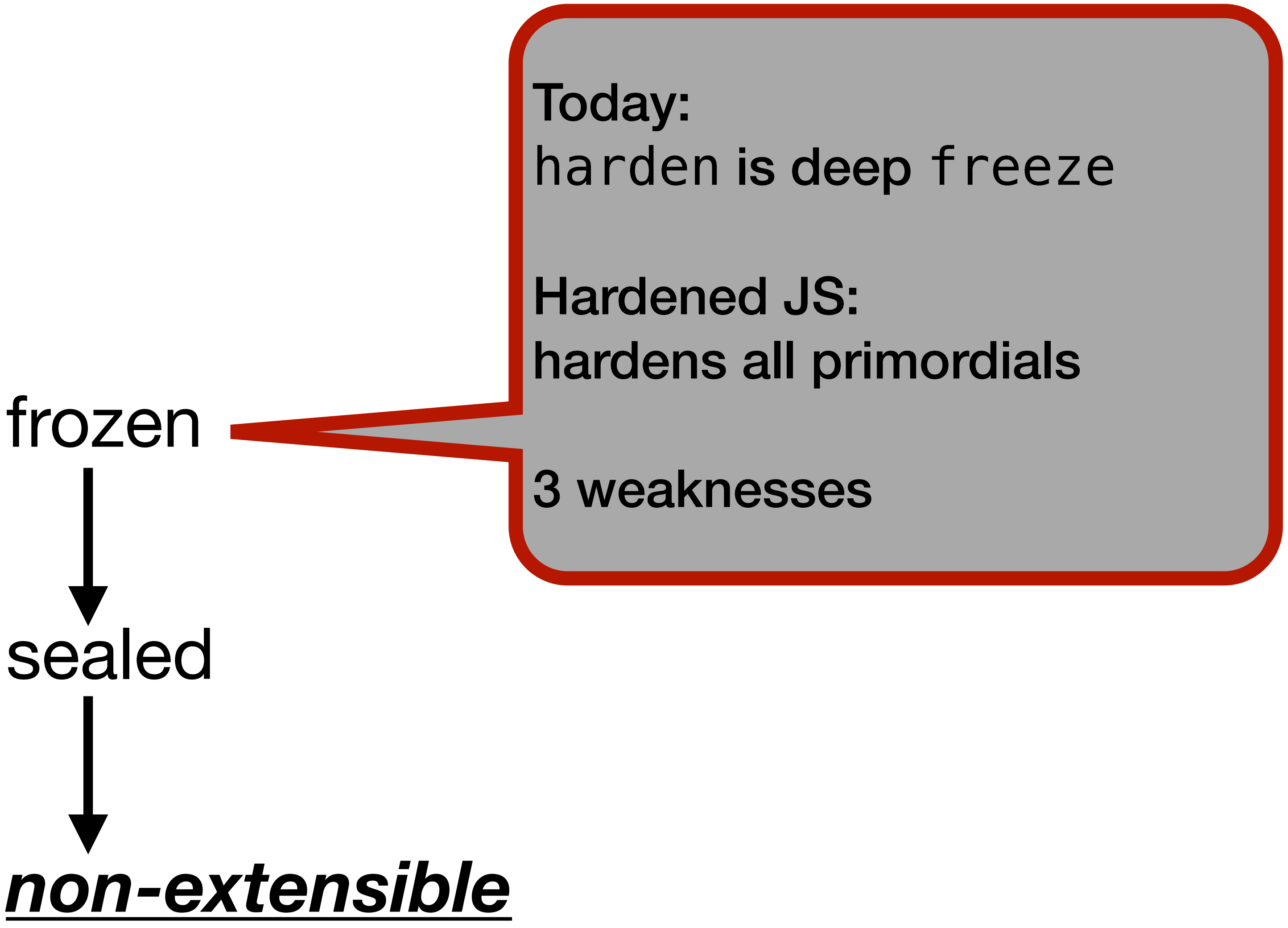
non-extensible

`isFrozen`

`isSealed`

`!isExtensible`

Current Integrity "Levels"



frozen

sealed

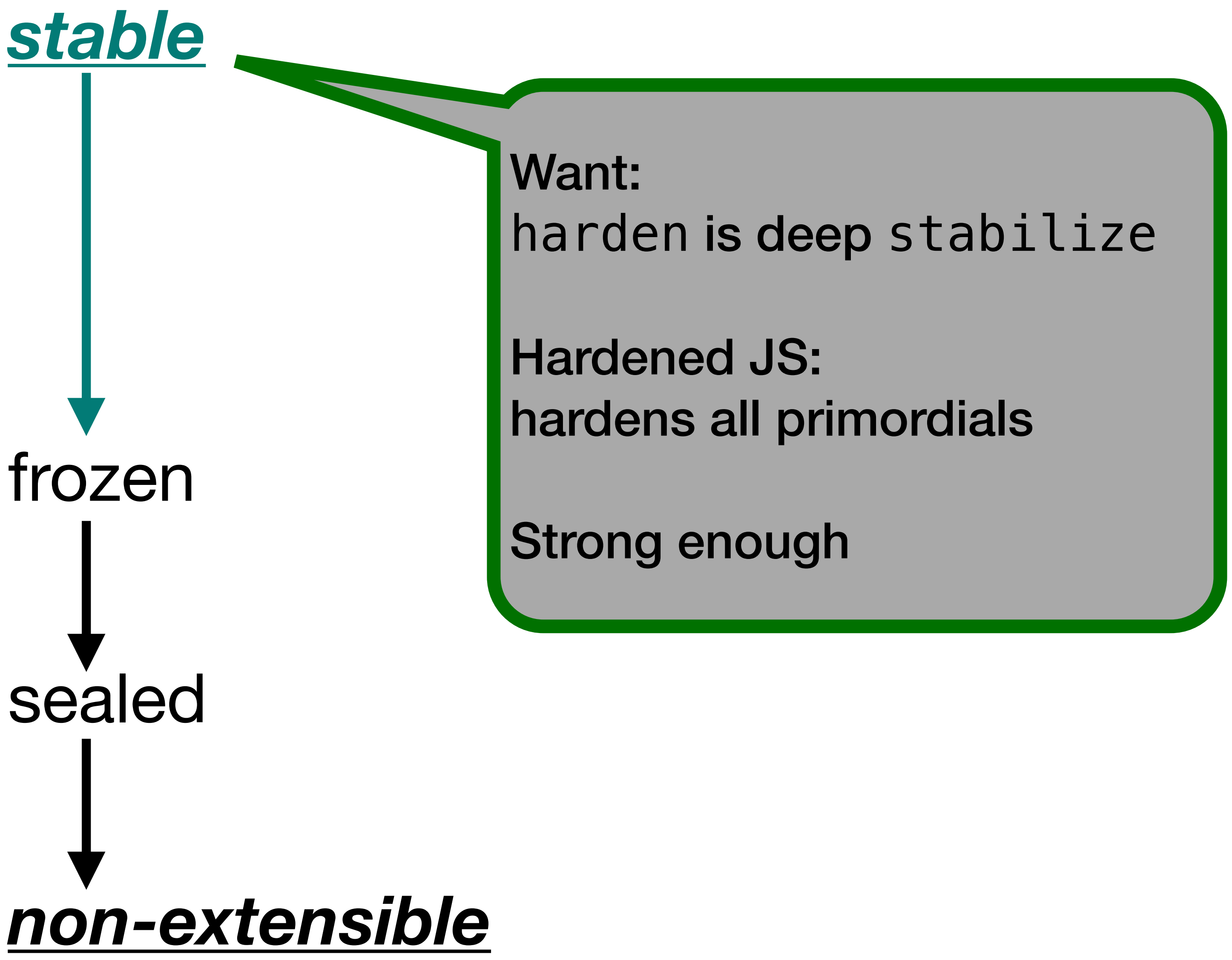
non-extensible

Today:
`harden` **is deep** `freeze`

Hardened JS:
hardens all primordials

3 weaknesses

First Try



stable

frozen

sealed

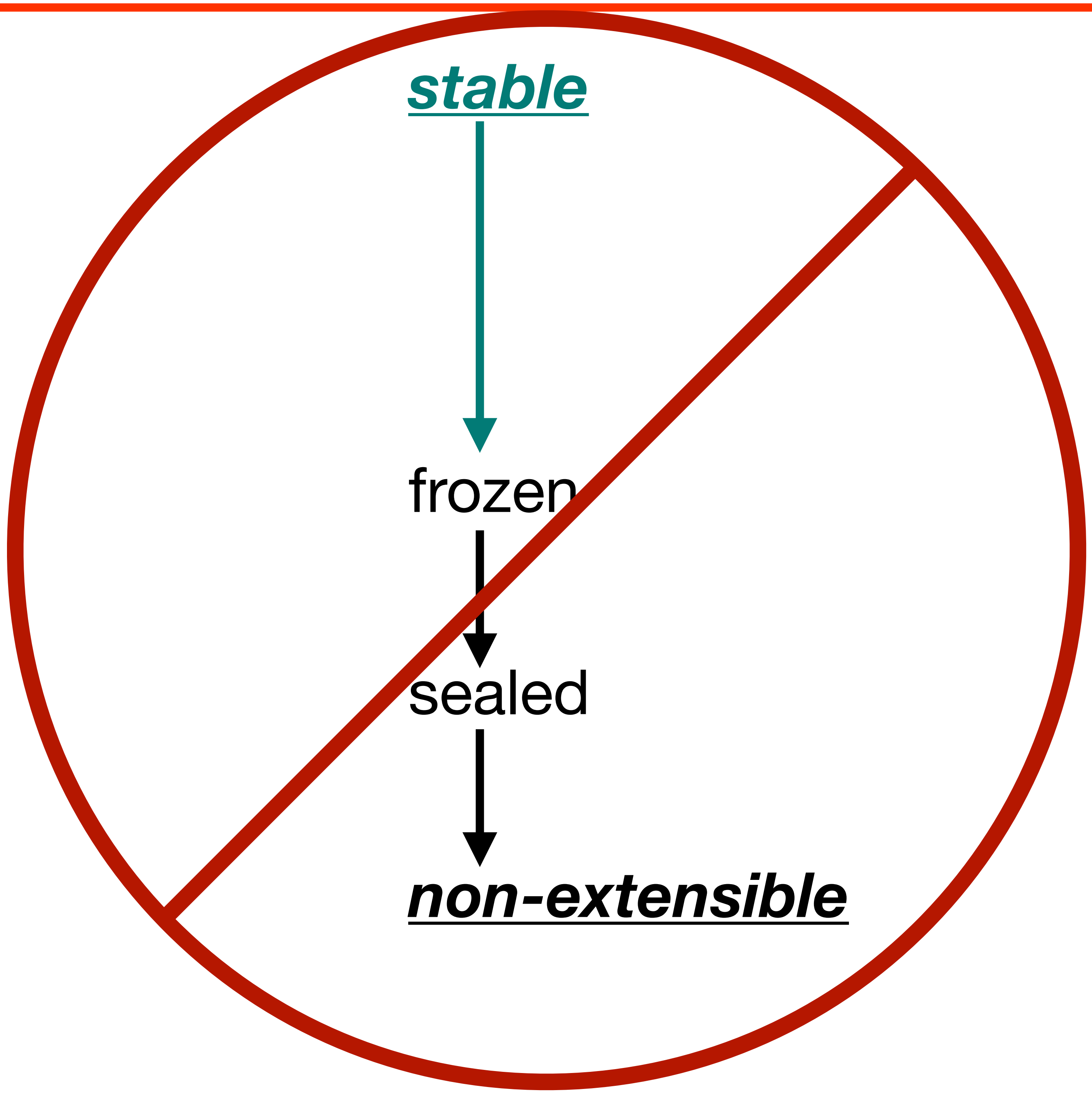
non-extensible

Want:
`harden` is deep `stabilize`

Hardened JS:
hardens all primordials

Strong enough

First Try. But Structs Cannot be Frozen



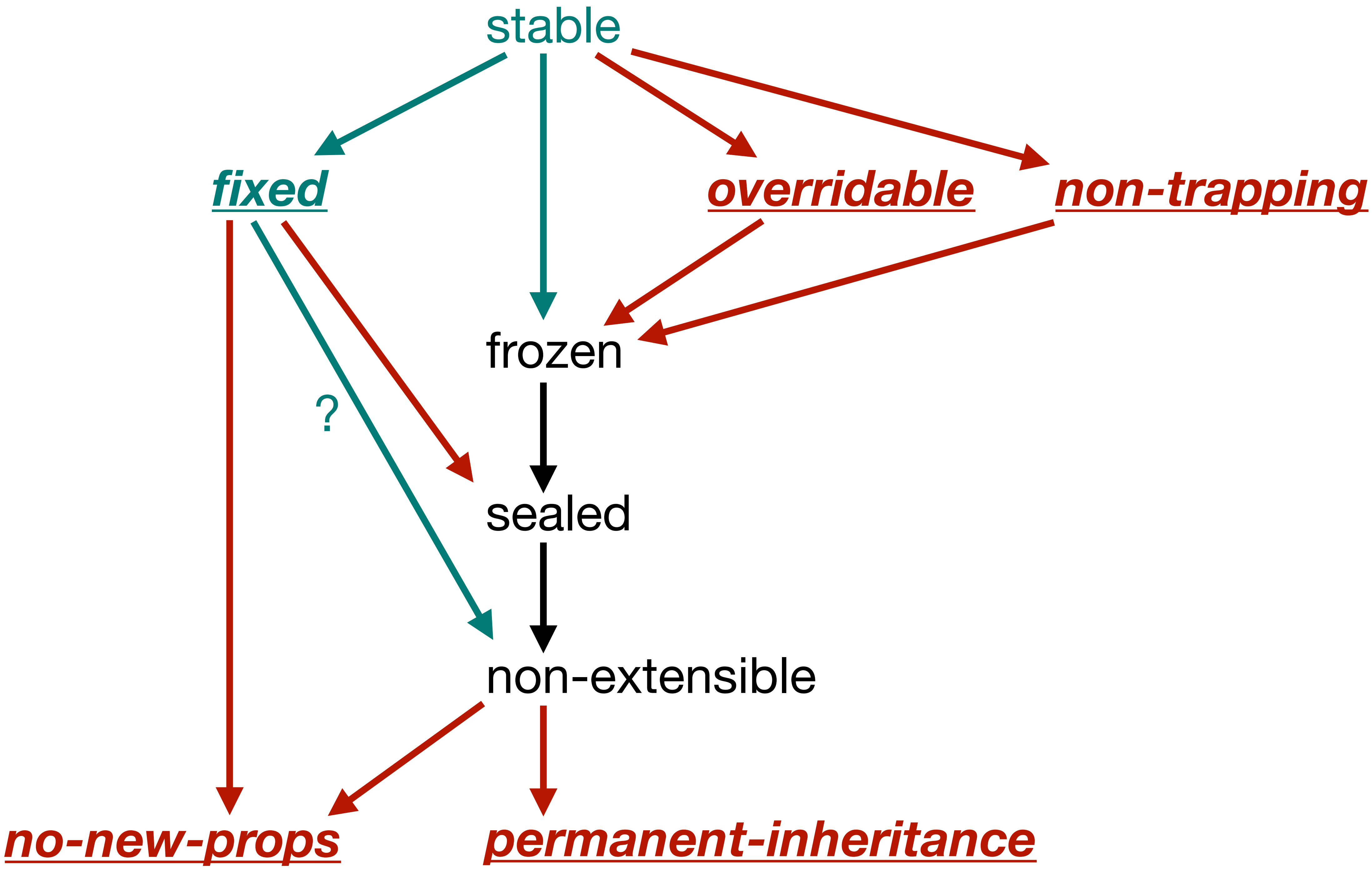
stable

frozen

sealed

non-extensible

All Considered Integrity “Traits”



stable

fixed

overridable

non-trapping

frozen

?

sealed

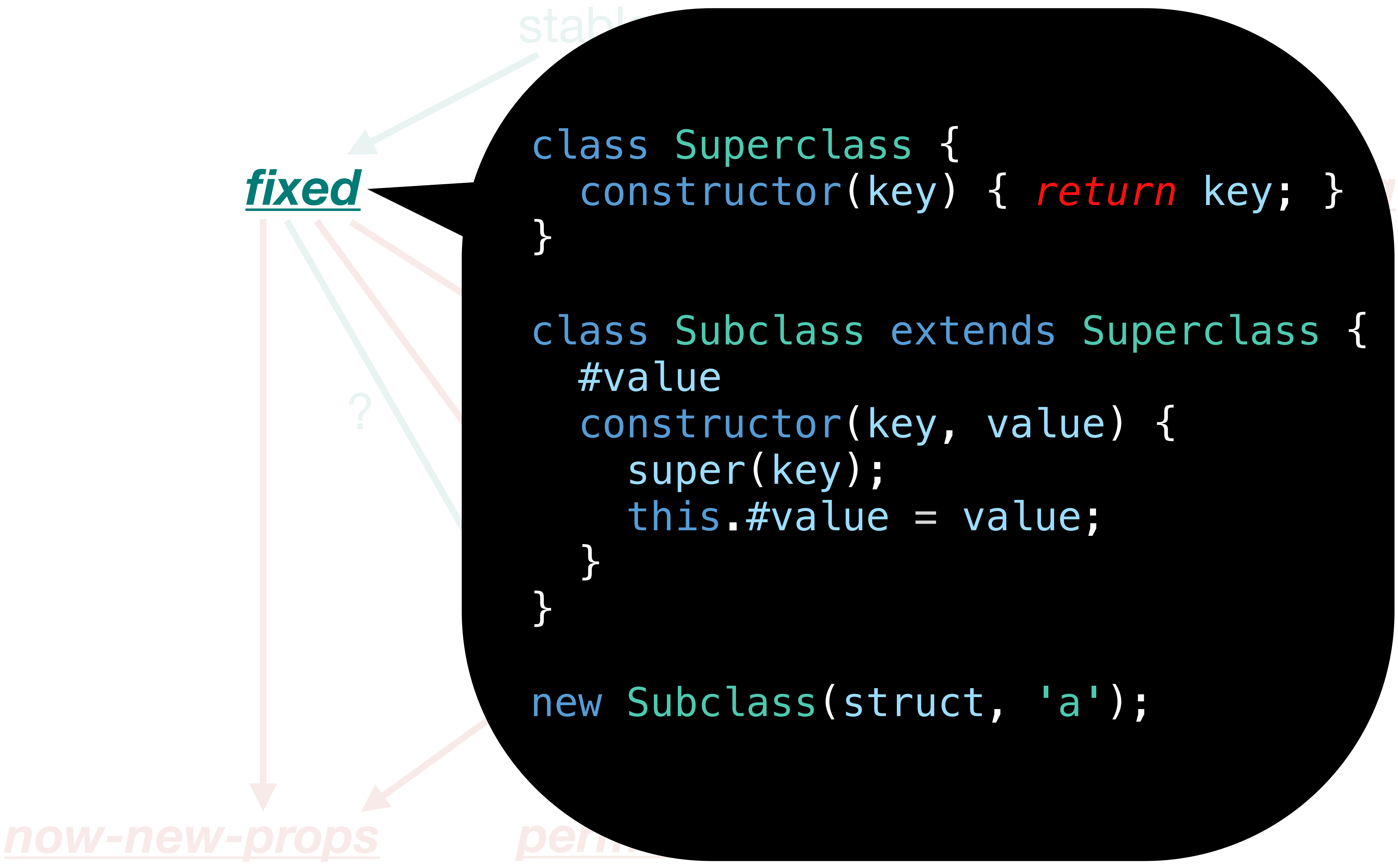
non-extensible

no-new-props

permanent-inheritance

fixed

Mitigate return-override mistake



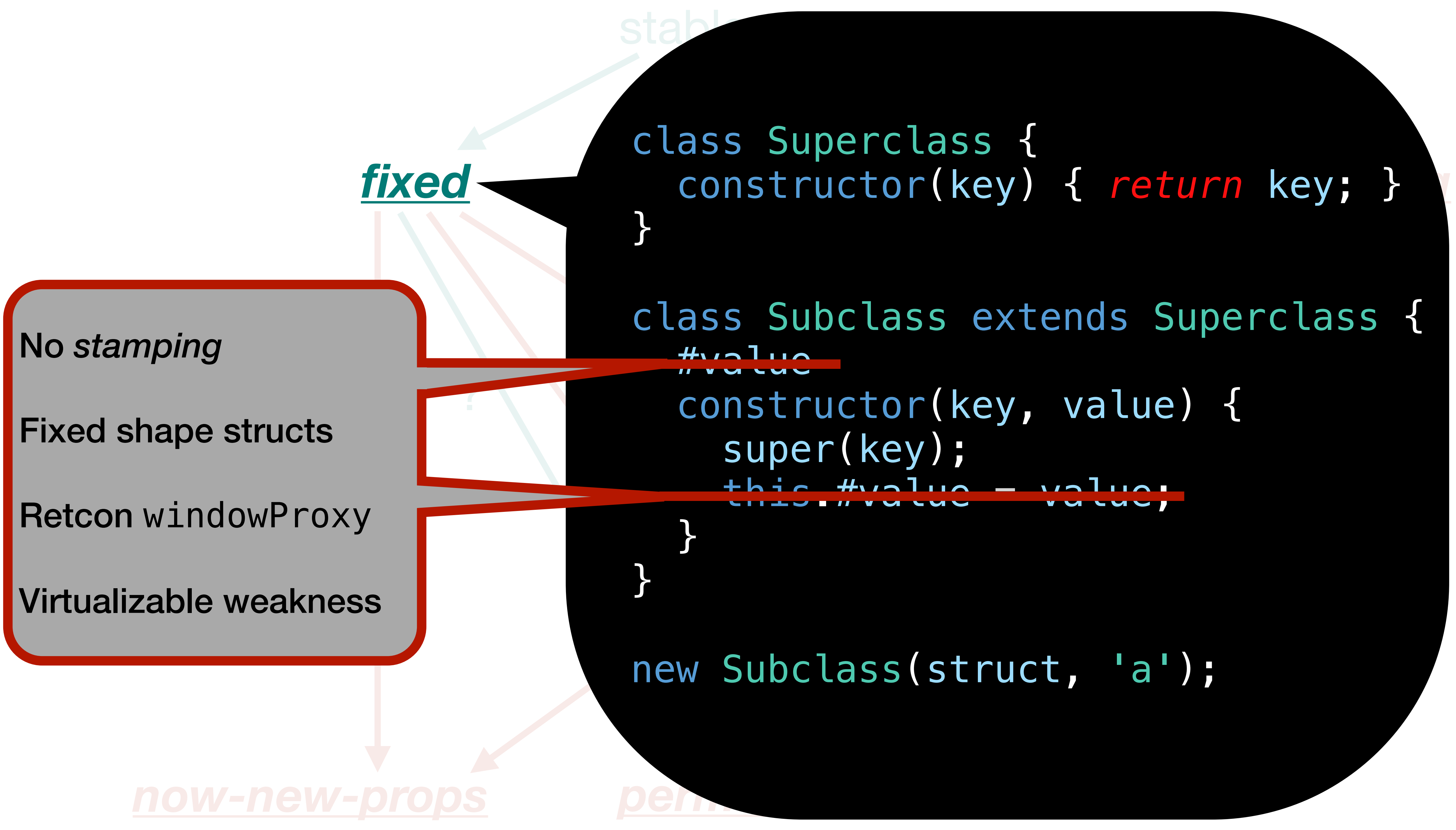
```
class Superclass {
  constructor(key) { return key; }
}

class Subclass extends Superclass {
  #value
  constructor(key, value) {
    super(key);
    this.#value = value;
  }
}

new Subclass(struct, 'a');
```

fixed

Mitigate return-override mistake



```
class Superclass {
  constructor(key) { return key; }
}

class Subclass extends Superclass {
  #value
  constructor(key, value) {
    super(key);
    this.#value = value;
  }
}

new Subclass(struct, 'a');
```

No *stamping*

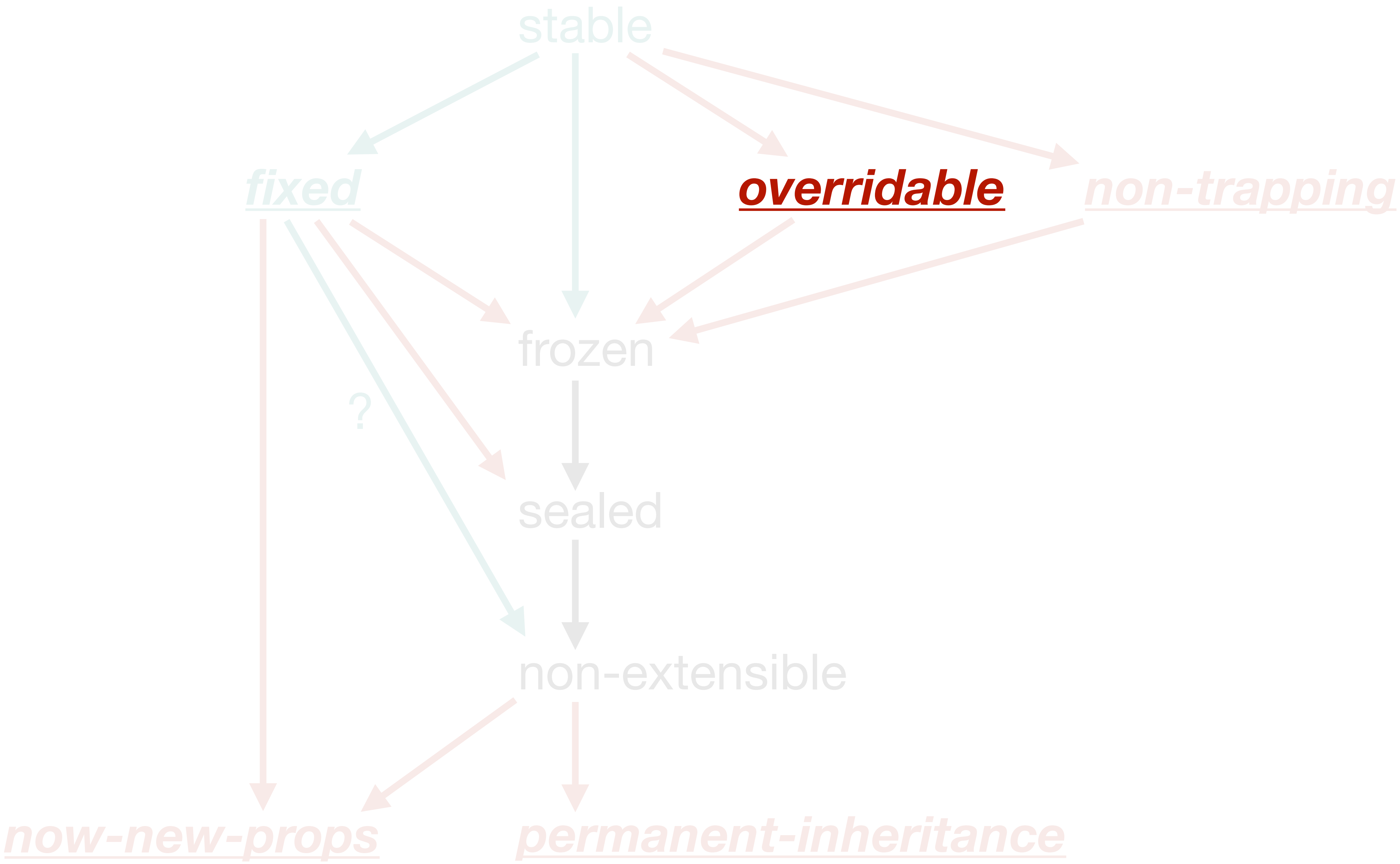
Fixed shape structs

Retcon `windowProxy`

Virtualizable weakness

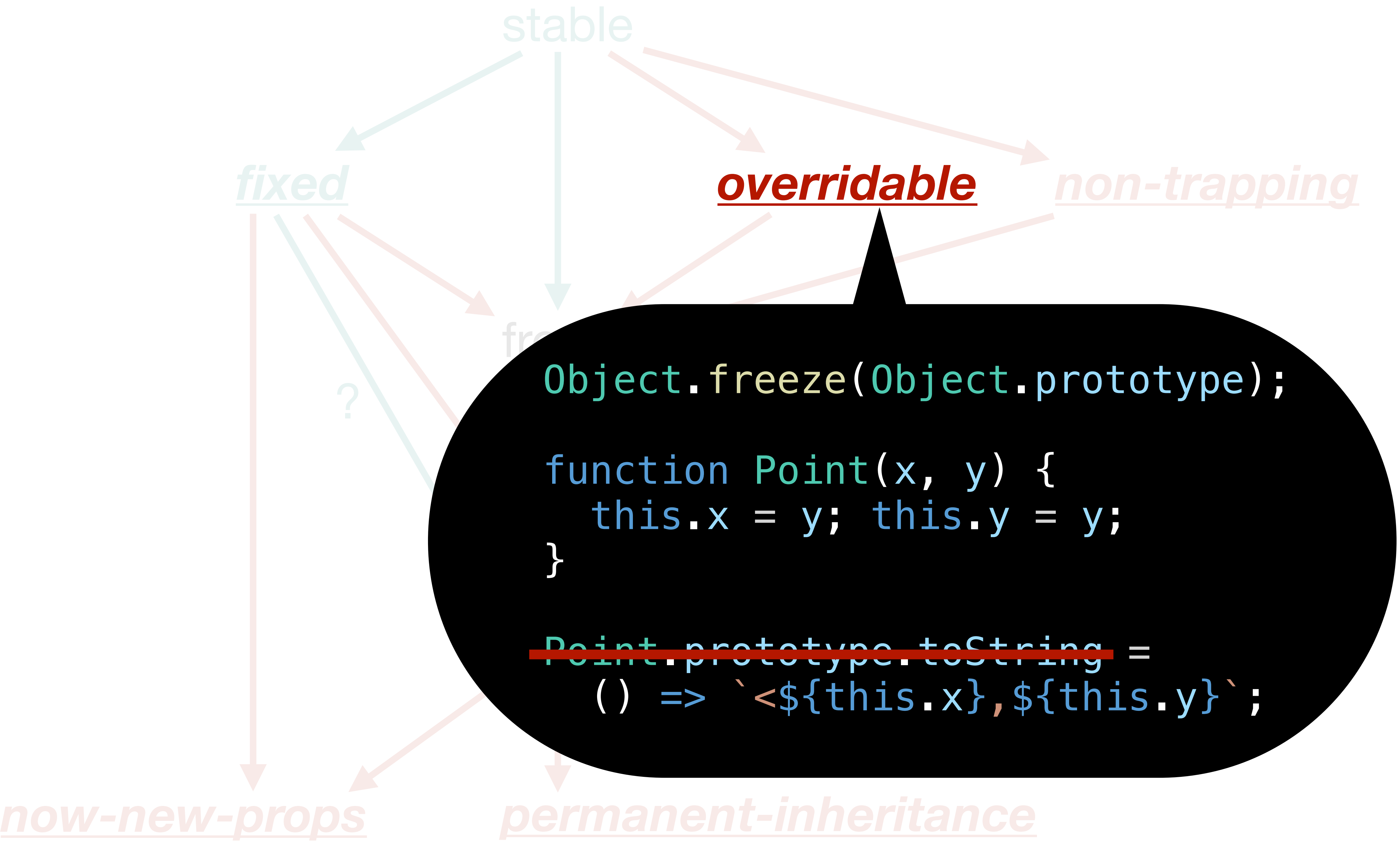
overridable

Mitigate assignment-override mistake



overridable

Mitigate assignment-override mistake



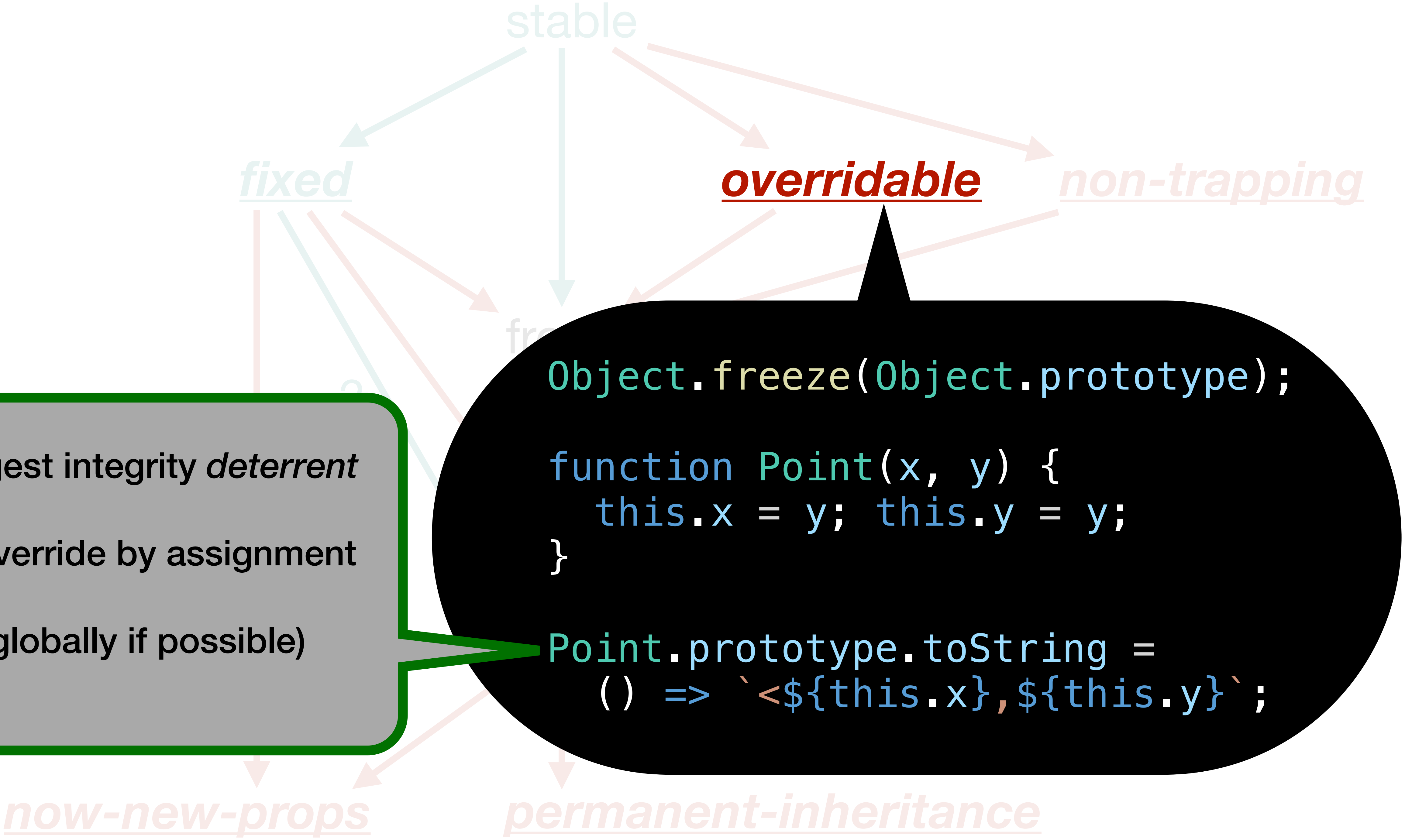
```
Object.freeze(Object.prototype);

function Point(x, y) {
  this.x = y; this.y = y;
}

~~Point.prototype.toString~~ =
  () => `<${this.x},${this.y}`;
```

overridable

Mitigate assignment-override mistake



```
Object.freeze(Object.prototype);

function Point(x, y) {
  this.x = y; this.y = y;
}

Point.prototype.toString =
  () => `<${this.x},${this.y}`;
```

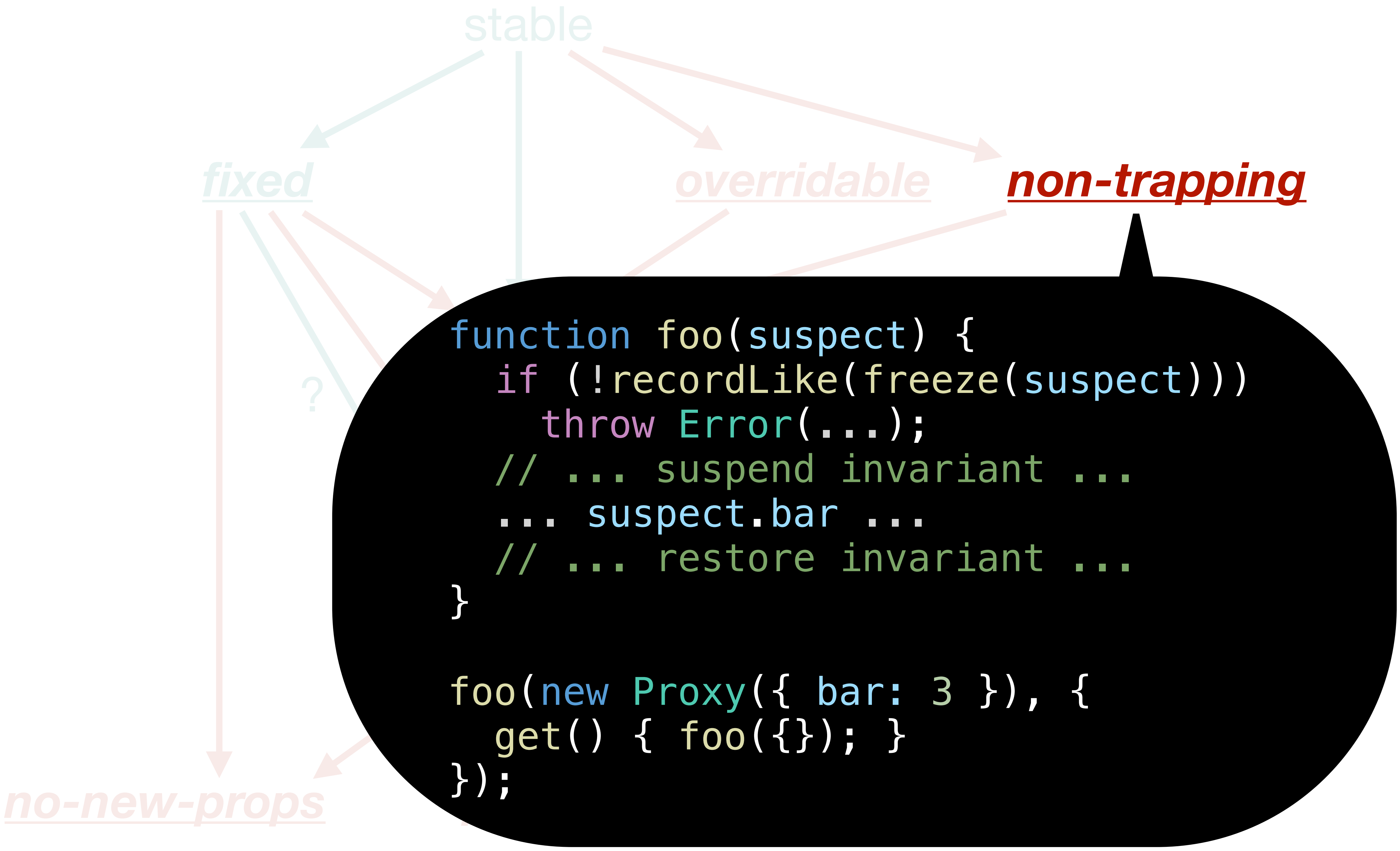
Fix biggest integrity *deterrent*

Allow override by assignment

(better globally if possible)

non-trapping

Mitigate proxy reentrancy hazards

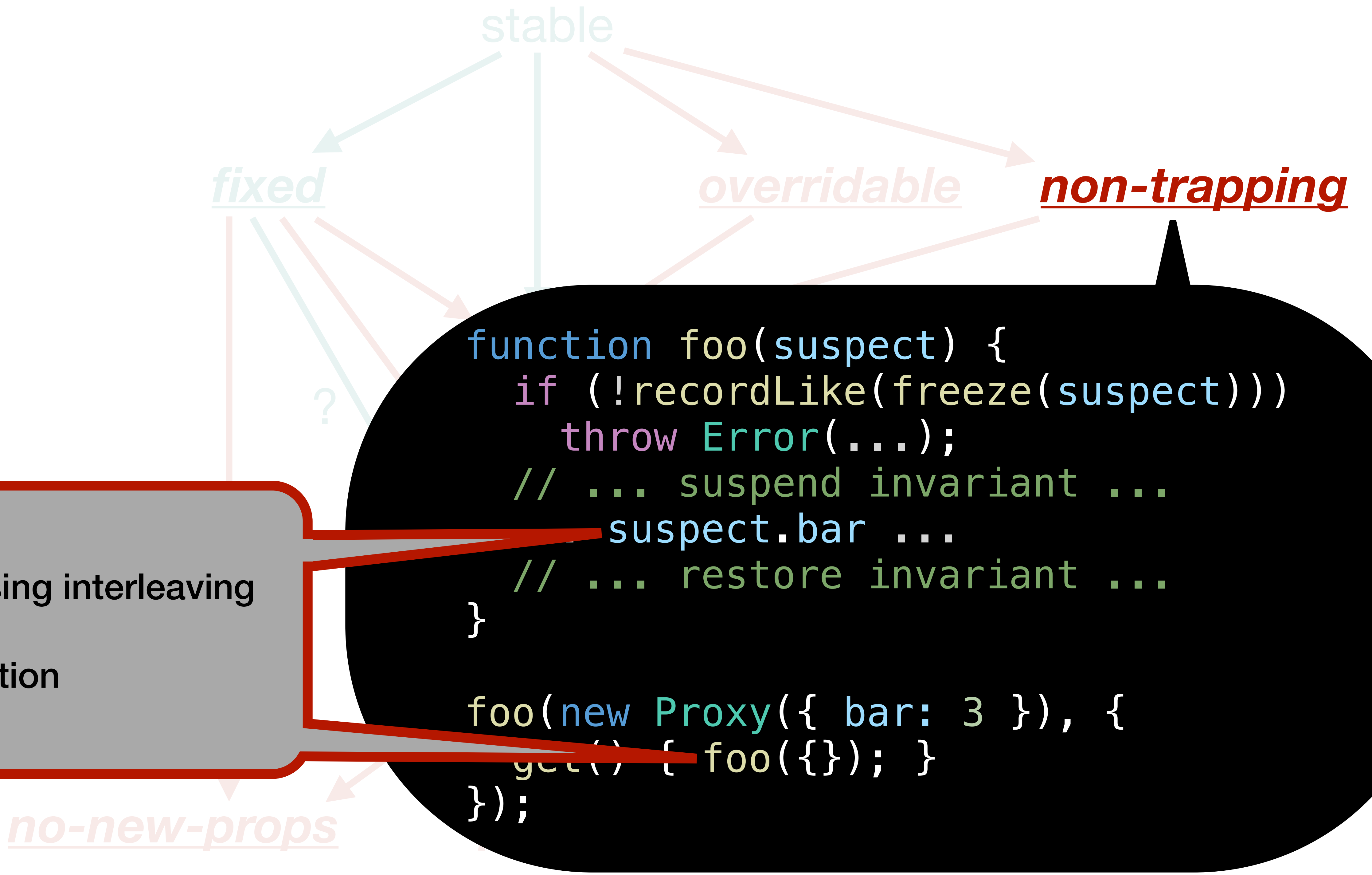


```
function foo(suspect) {
  if (!recordLike(freeze(suspect)))
    throw Error(...);
  // ... suspend invariant ...
  ... suspect.bar ...
  // ... restore invariant ...
}

foo(new Proxy({ bar: 3 }), {
  get() { foo({}); }
});
```

non-trapping

Mitigate proxy reentrancy hazards



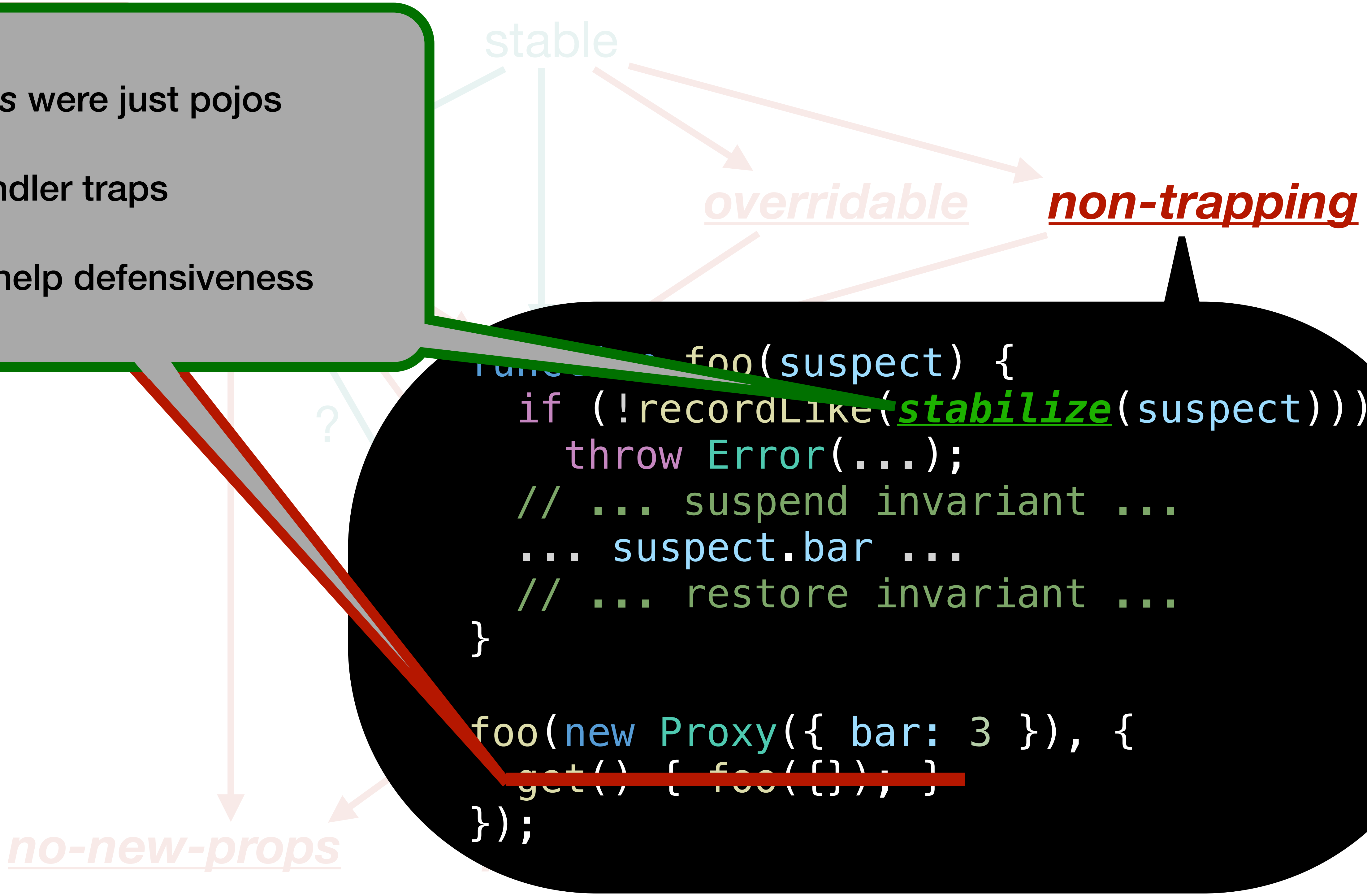
```
function foo(suspect) {
  if (!recordLike(freeze(suspect)))
    throw Error(...);
  // ... suspend invariant ...
      suspect.bar ...
  // ... restore invariant ...
}

foo(new Proxy({ bar: 3 }), {
  get() { foo({}); }
});
```

surprising interleaving

corruption

non-trapping
Mitigate proxy reentrancy hazards



records were just pojos

No handler traps

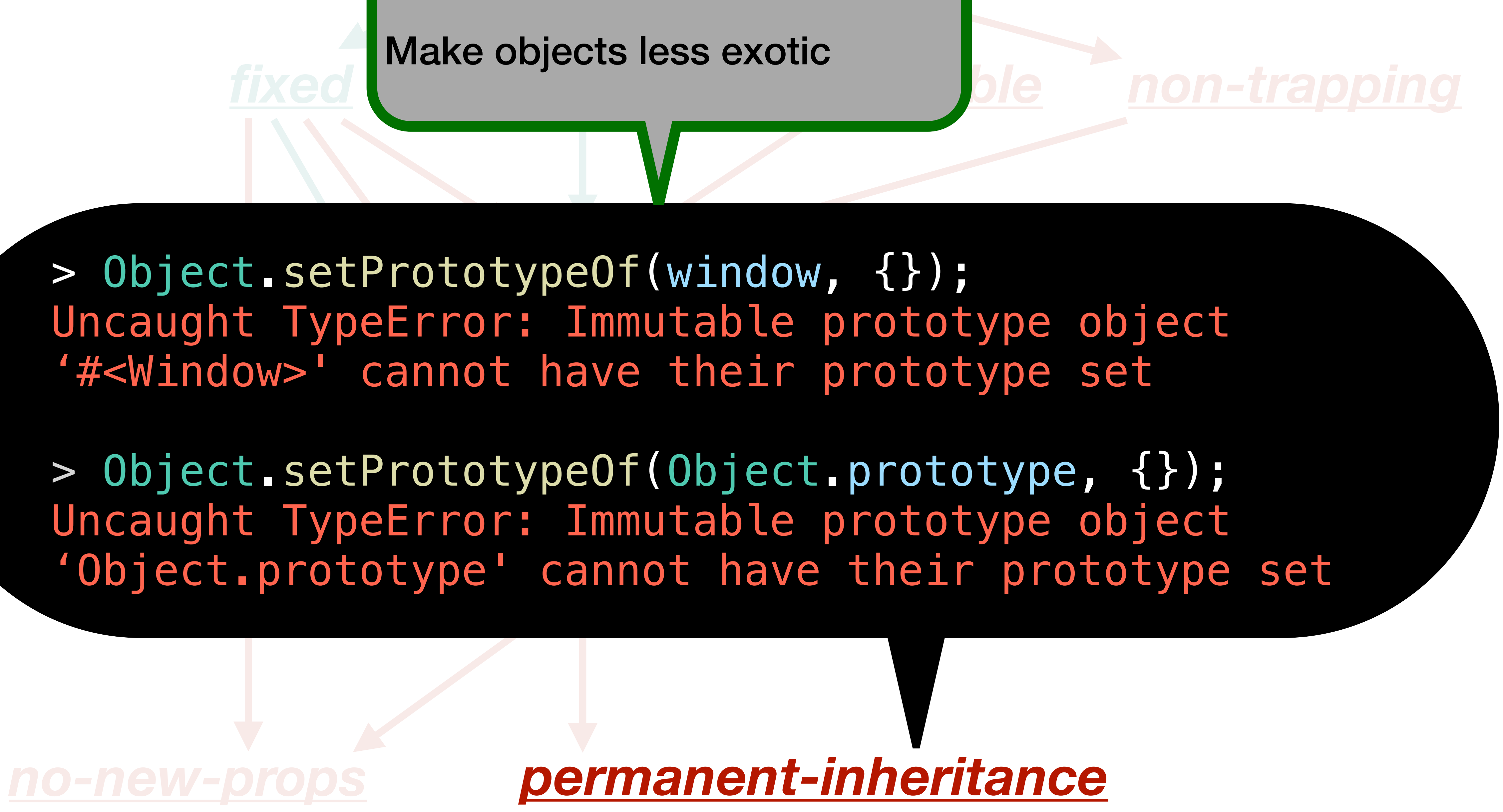
pojos help defensiveness

```
function foo(suspect) {
  if (!recordLike(stabilize(suspect)))
    throw Error(...);
  // ... suspend invariant ...
  ... suspect.bar ...
  // ... restore invariant ...
}

foo(new Proxy({ bar: 3 }), {
  ~~get() { foo({}); }~~
});
```

permanent-inheritance

Retcon `windowProxy`, `Object.prototype`



Too much magic

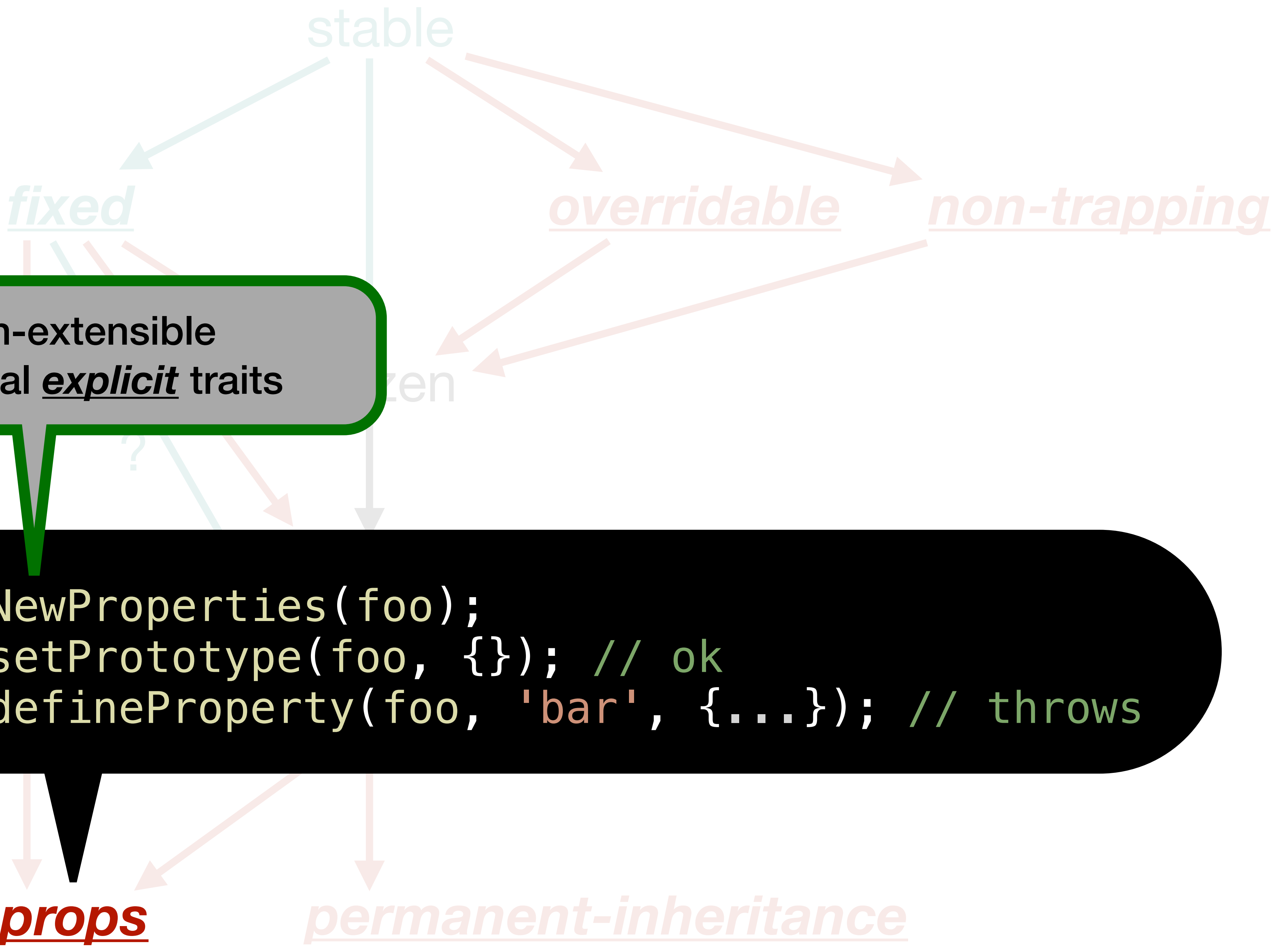
Make objects less exotic

```
> Object.setPrototypeOf(window, {});
Uncaught TypeError: Immutable prototype object
‘#<Window>' cannot have their prototype set

> Object.setPrototypeOf(Object.prototype, {});
Uncaught TypeError: Immutable prototype object
‘Object.prototype' cannot have their prototype set
```

no-new-props

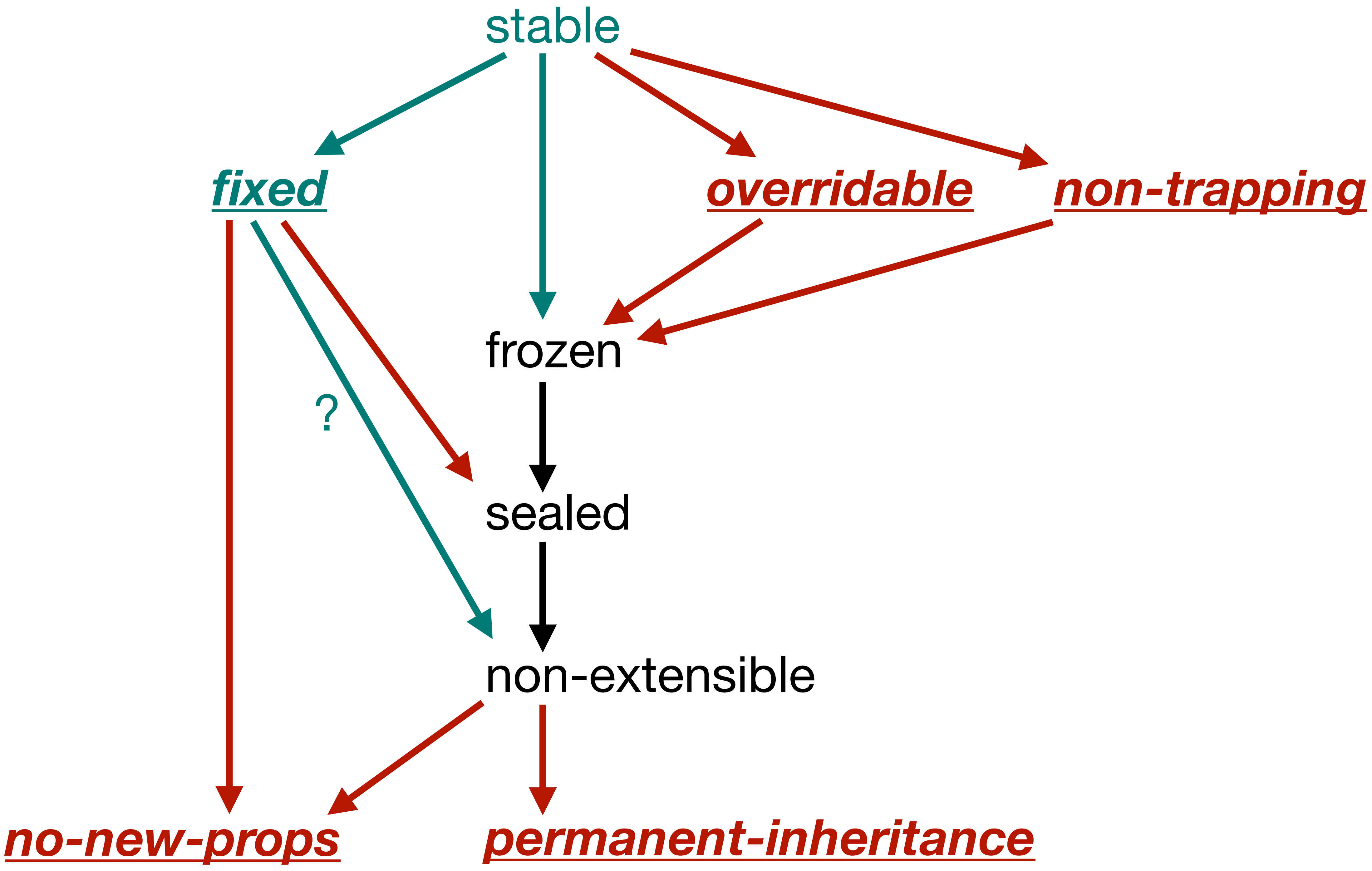
Rest of non-extensible unbundling



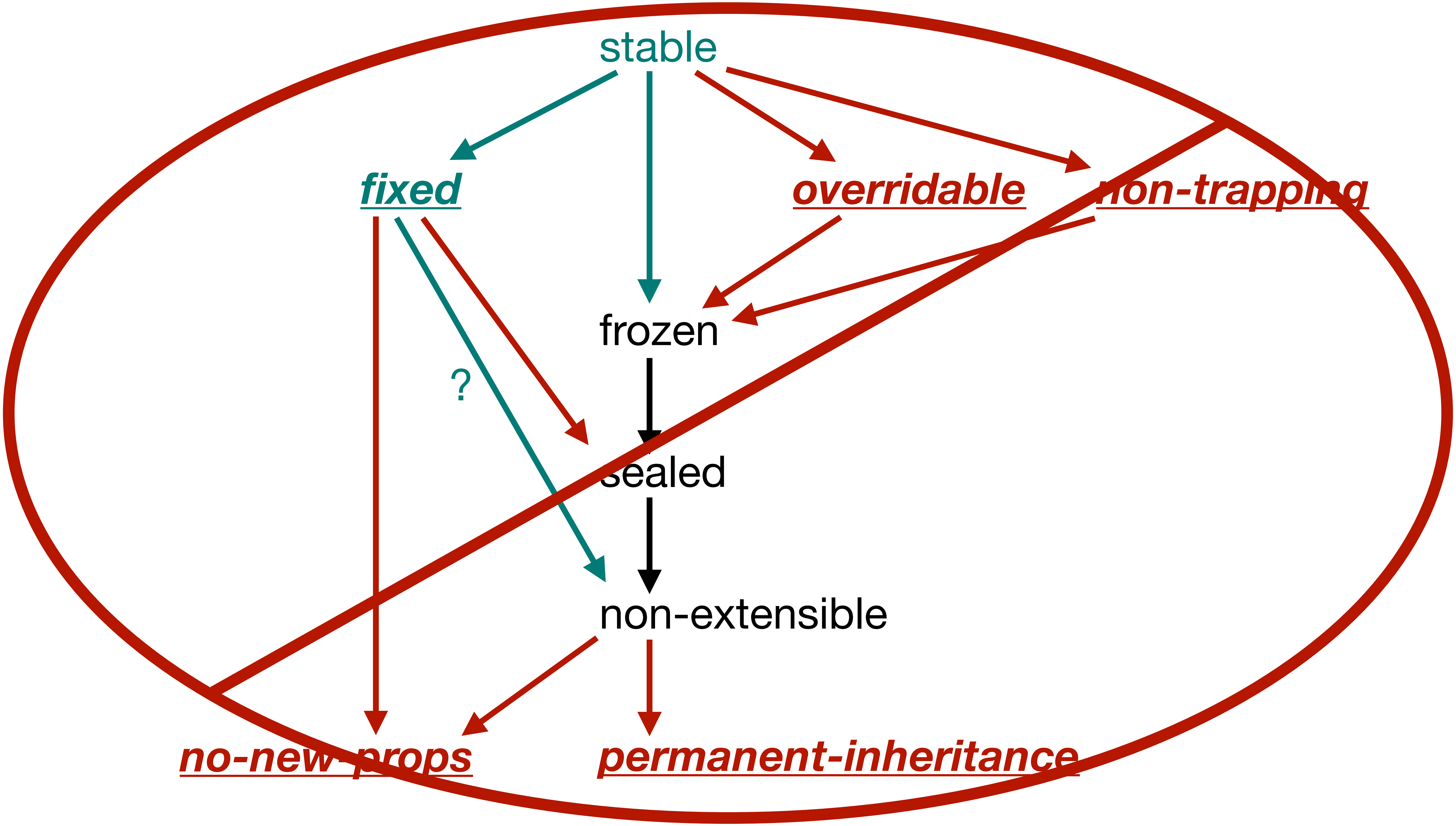
Unbundle non-extensible
into orthogonal ***explicit*** traits

```
preventNewProperties(foo);
Object.setPrototype(foo, {}); // ok
Object.defineProperty(foo, 'bar', {...}); // throws
```

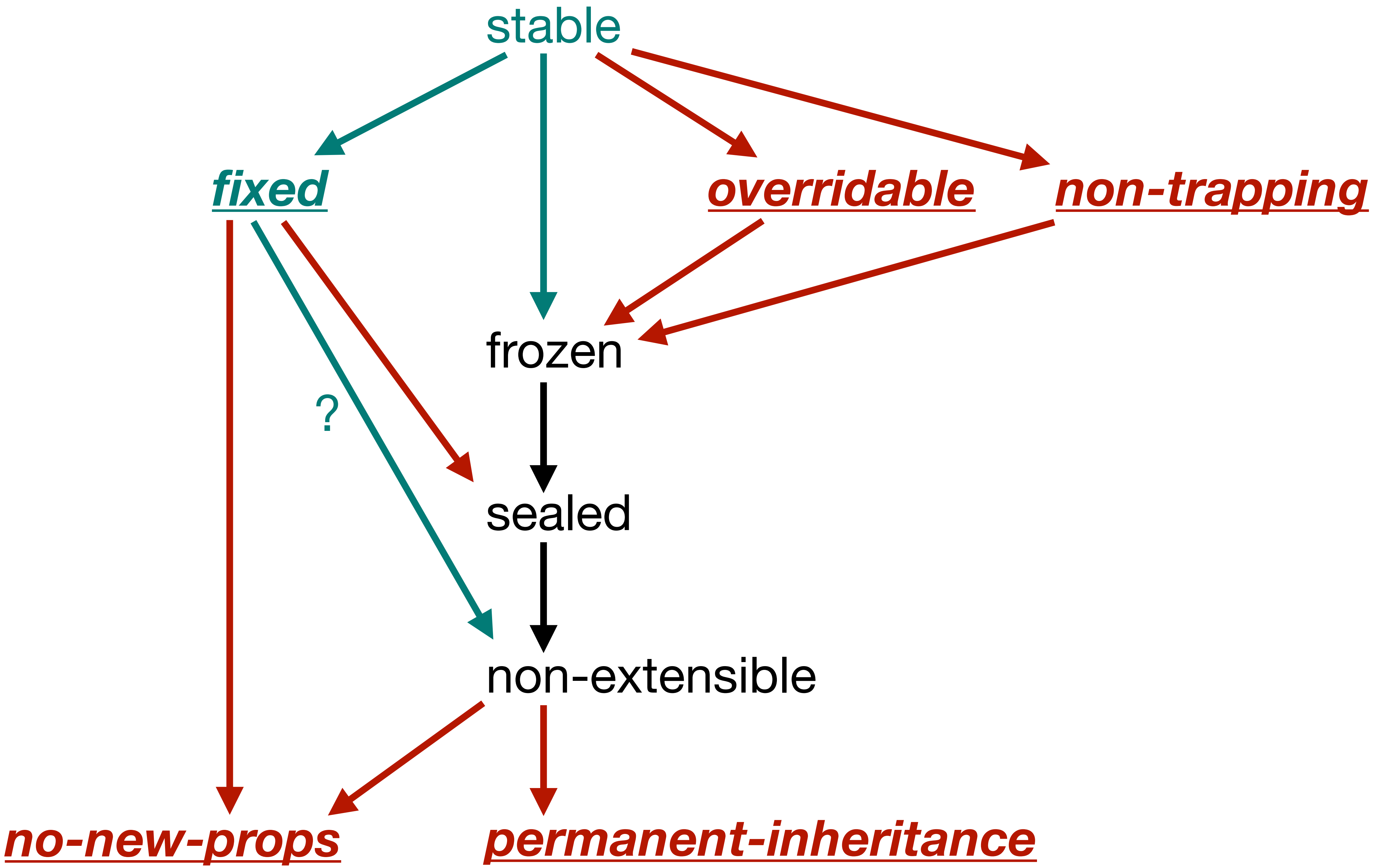
Maximally Unbundled Integrity “Traits”



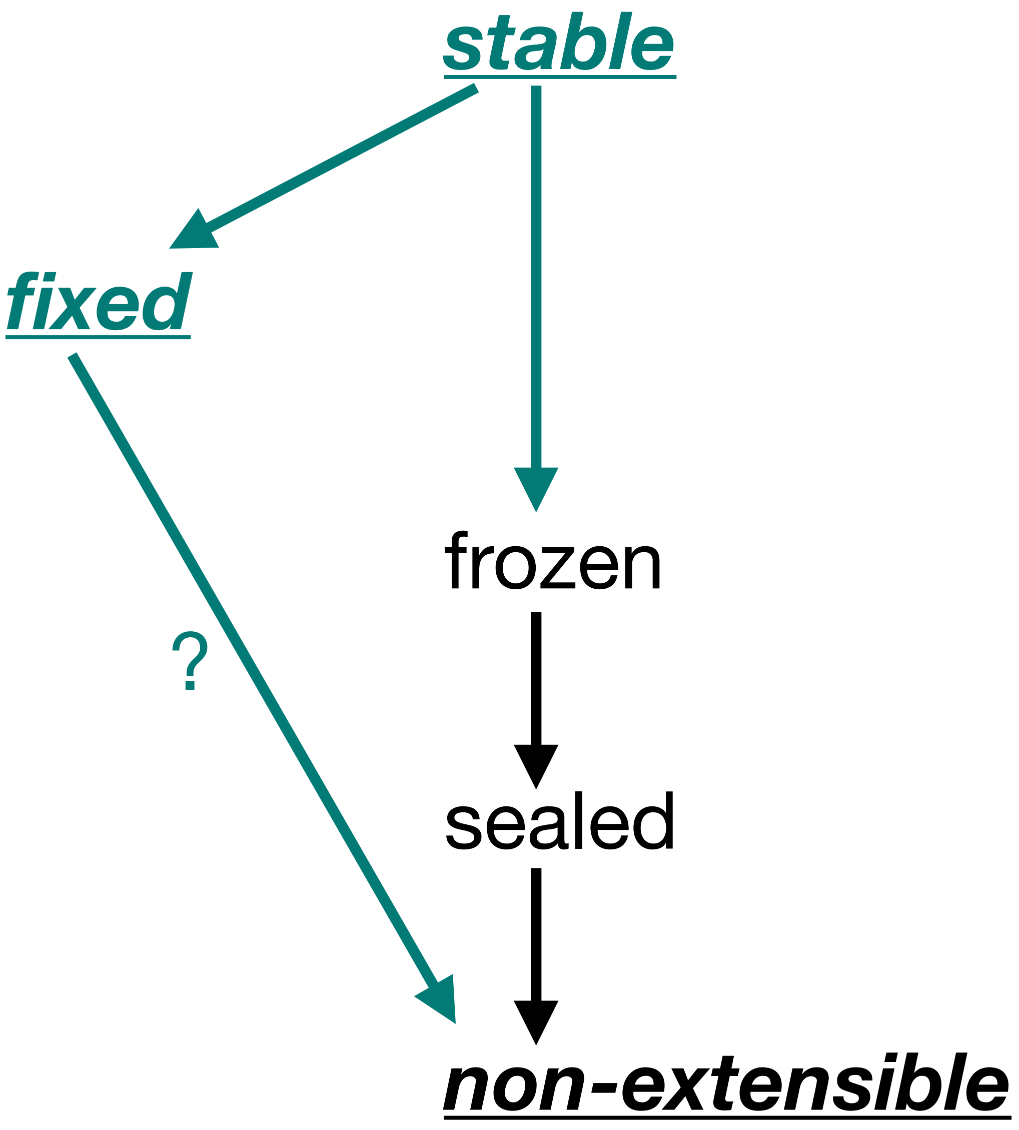
10 New Proxy Traps — Too many!



10 New Proxy Traps — Too many!



Favorite Proposed Integrity “Traits”


stable
fixed
frozen
?
sealed
non-extensible