eager-THiNG/vm-java 11 28 Aug 2006.

```
[ .name = x;
  .age  → now() - y;

  .ageIn year → year - y;

  .likes .red → .true;
  .likes _    → .false;
]
```

⇒

```
[ .name = x;
  .age  = now() - y;
  .ageIn → [[ year → year - y ]];
  .likes → [[ .red → .true;
              _    → .false; ]];
]
```

[ ] — bind self.
[[ ]] — do not bind self

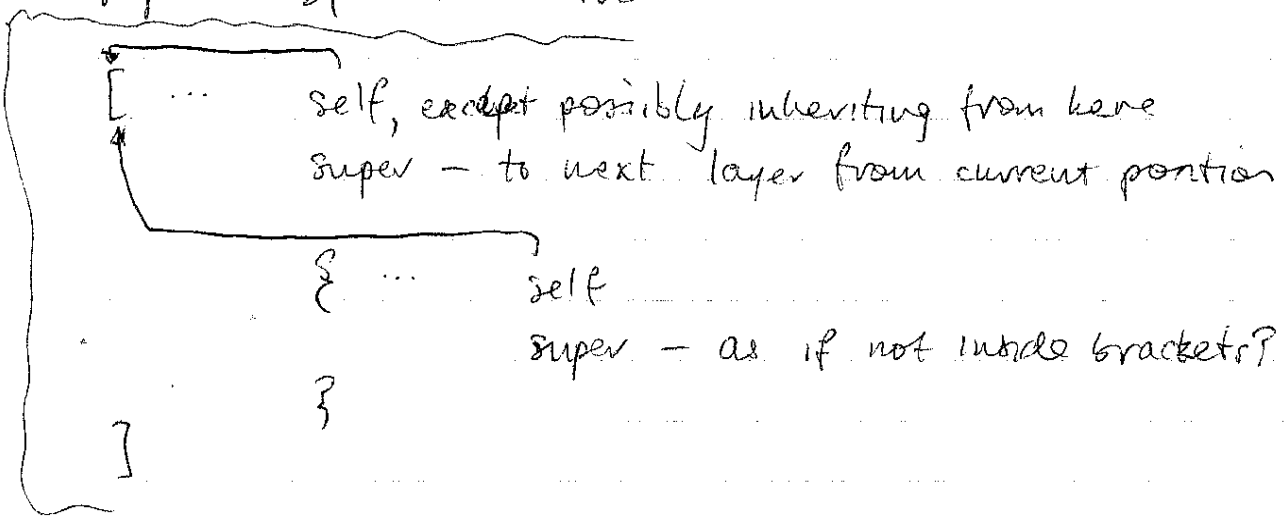
Trait operations: + / @ - ↑

+	sum	merge objects with no pattern overlap permitted
/	override	merge, with conflicts resolved to the left object.
@	rename	transforms a message on its way through. (does it transform on the way back?)
-	remove	matching delegate to DNU handler; otherwise as for override
↑	inherit	as for override, but exposes the RHS to supercalls

variable
+binding
rator rand (leftassociative)

[ ] object - binds self.
{ } closure - does not bind self.

self - actual value.
super - special receiver



Closures can't be trait-opped??

{} + x → [+v → {} v] + x

{} ↑ x → [+v → {} v] ↑ x

or, more properly,

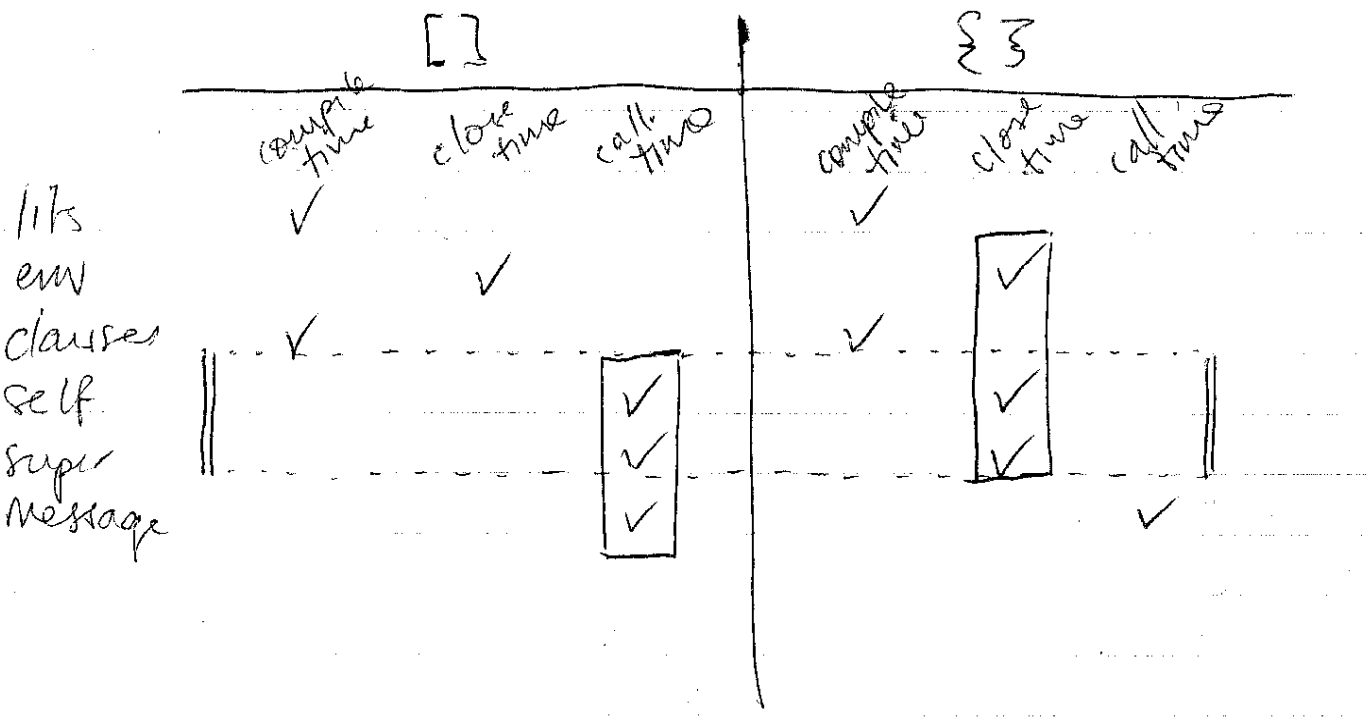
(+c = {} ; [+v → c v] ↑ x)

[] → more late-bound, with open recursion and dynamic self + super

{} → less late-bound, with no open recursion and static self + super.

[] : object
- static: literals, environment closed over
- dynamic: clauses, self, super, message

{} : closure
- static: literals, environment closed over, clauses
- dynamic: self, super, message



The only problem is coming up with a convenient syntactic shorthand for every closure (block).

{(...)}

[p p → v ; ⋯]

{ p p → v ; ⋯ }

{ v }

{{ v }}

ok because closures can appear in patterns?

{| p p → v ; ⋯ |}

(| p p → v ; ⋯ |)

[| p p → v ; ⋯ |]