# Notes - 5/16: SENT

Monday, May 16, 2011 3:01 PM

- Type inference questions for implementation

```
var v = new Circle();
var x = v.radius;
v = new Shape();
```

The above should type check, the type of v is given at it's declaration.

Comes down to "var v;" – what type does this get?

- **Answer: any**

Thought: warning level to warn here? Also on parameters with no type annotation?

Circular dependencies in inference:
Algorithm:

- Mark functions as in-progress or finished for type checking
- If in-progress and reached:
    - Error? Quite strict
    - Var? Types disappear silently, hard to tell why.
    - Poor-man's unification: all functions in the chain return "unknown", iterate type checking with these to fixed point, error if no progress

**Decision: (temporary) Let's start with error and see how painful it is.**

What's the type of "[x,y,z]"?

Shape,Circle,Shape -> Shape[] (

Can start simple, they must all be the same?

Intersection:

- Classes: only exists if types are compatible

```
Any
|- string
|- number
|- {}
  |- IX
  |- IY
```

function foo() => something ? a : b;

Question: What interface names do we pick?
Answer: Prefer named when available, pick the "frist"?

Rules:

- Pick the lowest compatible type
- Report error if the type isn't one of the ones mentioned
- If there are multiple that have the lowest compatible type, pick a named one. If there are multiple named, pick the first.

```
[ix, cx] => ix[]
[shape, circle, square] => shape[]
[circle, square] => error
[anonix, ix] => ix[]
```

A class is an interface with an additional member branding it as unique from other structurally compatible classes.

**Statement: "Any is contagious – but it doesn't come out of nowhere. "**

Array convertability rules:
- **Pointwise assignment compatibility with declared element type**
- But can't verify
- In debug mode, we could insert validation – but there will be some definite limit here
    - Checking arrays is very expensive

Rest of backlog:
- Dotting into typed objects (dynamically)
- Coalescing operator (??)
- Null propagation
- Possible operator to easily convert to any
- What is the type of null?

Proposal:

foo!x – equivalent to (foo as any).x

(foo as any).x // correct
foo["x"] = 5; // need to discuss

Grammar: This is an infix

Decision: No ! yet!

- Is and as, possible T(x)

(foo as any) // syntax for casts, no verification, syntax error on illegal conversions, two of these allows anything -> anything

(foo is ICircle) // not allowed – can't check for interfaces
(foo is int[]) // not allowed
(foo is number) // okay
(foo is Circle) // okay?

Question: What about ":" instead of "as"?
Question: Should "is" be library? No for now

```
Function foo(x:any) {
  If(x is ICircle) {
    Var circ  = x as any;
    Console.log(Circ.radius);
  }
}
```

- Top level scope?
- Uninitialized variables/inference/defaults
- O[x] with typed objects
- Enums
- Global module
- Type compatibility rules
- Foreach
- Class-parameters as privates and inheritance
- Function type compatibility (omit parameters, more parameters, varargs, etc.)
- Any -> number (what gets emitted)

**From:** Luke Hoban
**Sent:** Monday, May 16, 2011 8:13 AM
**To:** Strada Design Team
**Subject:** Strada Design Meeting Agenda - 5/16/2011

[Note – we'll be starting 30 mins late today at 10:30]

We've got the following design topics on our backlog – we'll try to make some progress on these today.
- Dotting into typed objects (dynamically)
- Is and as, possible T(x)

- Top level scope?
- Uninitialized variables/inference/defaults
- O[x] with typed objects
- Possible operator to easily convert to any
- What is the type of null?
- Enums
- Coalescing operator (??)
- Null propagation
- Global module
- Type compatibility rules
- Foreach

We've also got a few higher-level topics that we may want to tackle today if we have time:

- Details of path to external and internal CTP
- Type checking rules for expressions and statements
- Typing of base class libraries
- Finish design of modules (import syntax,ALIASING, remote code loading, dependency resolution, etc.)
- Feedback on ES standards strategy memo Amanda sent around, or details we want to push into that discussion (like the int conversion)

Anything else?

Luke