Notes On Memory

memory is like a big suburb full of houses with addresses, and a *variable* is an easy-to-remember/easy-to-read label like `Susie Derkins's house`, which translates to `220 Hastings St, Cincinnati, OH 45219`. The *value* is whatever's in the house.

You can have more than one label for a house.

You can say `The biggest green Victorian in Cincinnati` and still mean `220 Hastings St, Cincinnati, OH 45219` which still uniquely identifies the same actual house, which still contains whatever it contains.

Aka, two variables (labels) can point at the same memory address, which contains whatever it contains.

If that house contains a *scalar value* like a number, string, etc., then when you send a messenger over to pick up a package from there, you just get that one thing back.

If that house contains an *object value* like an `Array` or an `Object`, then when you send a messenger over to pick up a package from there, you get a box back. You have to open that box to get at what's inside -- the usable values.

There might be other boxes inside that first box you get back (2-dimensional arrays, objects with properties that point at other objects) ...

But those boxes (`Arrays` and `Objects`) ultimately are useless without eventually getting scalar values like numbers and strings.

Related example: even if Amazon ships you a big box that has a variety of other boxes in it, in the end you need to open them all up to get at the hairbrush, toothbrush, and mouthwash you ordered -- the specific *nouns* that matter to you.

Those nouns will inevitably be numbers, strings, and whatnot. A box full of boxes isn't too useful without those fundamental/simplest nouns.

That's it.

That's why `[1,2,3] === [1,2,3]` evaluates to `false` <-- Hey, they even look like boxes...