

Instance

klass: Klass

fields: Map<String, Object>

get(name: String): Object

set(name: String, value: Object): void

Fun

decl: AST

closure: Environment

call(interpr: Interpreter, args: List<Object>): Object

bind(instance: Instance): Fun

Klass

name: String

methods: Map<String, Fun>

call(interpr: Interpreter, args: List<Object>): Object

```
Note..

Object call(Interpreter interpr, List<Object> args) {
    Environment prev = interpr.env;
    interpr.env = new Environment(closure);
    for (int i=0; i<args.size(); i++) {
        interpr.env.define(decl.params().get(i).getText(), args.get(i));
    }
    interpr.eval(decl.block());
    interpr.env = prev;
}</pre>
```

```
Note..
Object call(Interpreter interpr, List<Object> args) {
   Instance inst = new Instance(this); return inst;
}
```

```
Note..

Object get(String name) {
   if (fields.containsKey(name)) {
      return fields.get(name);
   }

Fun method = klass.findMethod(name);
   if (method != null) return method.bind(this);

throw new RuntimeError(name, "Undefined property");
}
```