# the taming of the

# deftype

Baishampayan “BG” Ghose
@ghoseb

Clojure/West 2012


@ghoseb
helpshift.com
prefectionist.

I caught a dead mouse in the office the other day

@ghoseb

helpshift.com

# MOTIVATION


THE MYSTERY MACHINE


HYDE PARK
KENSINGTON GARDENS
The Serpentine
KENSINGTON ROAD
BAYSWATER ROAD
MARYLEBONE ROAD
OXFORD STREET
Round Pond
Lancaster Gate
Marble Arch
Bond Street
Knightsbridge
Hyde Park Corner
PALACE GA
31
30
22
21
3
2
19
1

# Maps


# Records

# Types

# TO SUMMARIZE

# deftype

Named Type

Full Control

Performance

Mutable Fields

Collection Abstractions

# defrecord

Named Type

Performance

HashMap Semantics

Collection Abstractions*

# maps

Every other use-case

# IN SHORT

# Use Maps!

Friday 16 March 12

# Dive into deftype

Friday 16 March 12

github Search... Explore Gist Blog Help

clojure / src / jvm / clojure / **lang**

| name | age | message | history |
|---|---|---|---|
| .. | | | |
| AFn.java | 15 days ago | CLJ-855: throw exceptions directly instead of wrapping them with RTE [bpsm] | |
| AFunction.java | 15 days ago | CLJ-855: throw exceptions directly instead of wrapping them with RTE [bpsm] | |
| AMapEntry.java | 15 days ago | CLJ-855: throw exceptions directly instead of wrapping them with RTE [bpsm] | |
| APersistentMap.java | 5 months ago | use hash consistent with equiv [richhickey] | |
| APersistentSet.java | 25 days ago | CLJ-895 obey contract for toArray return type [stuarthalloway] | |
| APersistentVector.java | 25 days ago | CLJ-895 obey contract for toArray return type [stuarthalloway] | |
| ARef.java | 15 days ago | CLJ-855: throw exceptions directly instead of wrapping them with RTE [bpsm] | |
| AReference.java | a year ago | get rid of checked exceptions [richhickey] | |
| ASeq.java | 25 days ago | CLJ-895 obey contract for toArray return type [stuarthalloway] | |
| ATransientMap.java | a year ago | get rid of checked exceptions [richhickey] | |
| ATransientSet.java | a year ago | get rid of checked exceptions [richhickey] | |
| Agent.java | a year ago | get rid of checked exceptions [richhickey] | |
| ArityException.java | a year ago | 397 better error message when calling macros with arity [mikehinchey] | |
| ArrayChunk.java | a year ago | get rid of checked exceptions [richhickey] | |

# Use the source, Luke!

IPersistentStack

ILookup

IPersistentMap

IPersistentCollection

Seqable

IObj

Associative

IPersistentSet

IFn

Counted

IPersistentVector

Indexed

Object

IMeta

IPersistentStack

ILookup

IPersistentMap

IPersistentCollection

Seqable

IObj

Associative

IPersistentSet

IFn

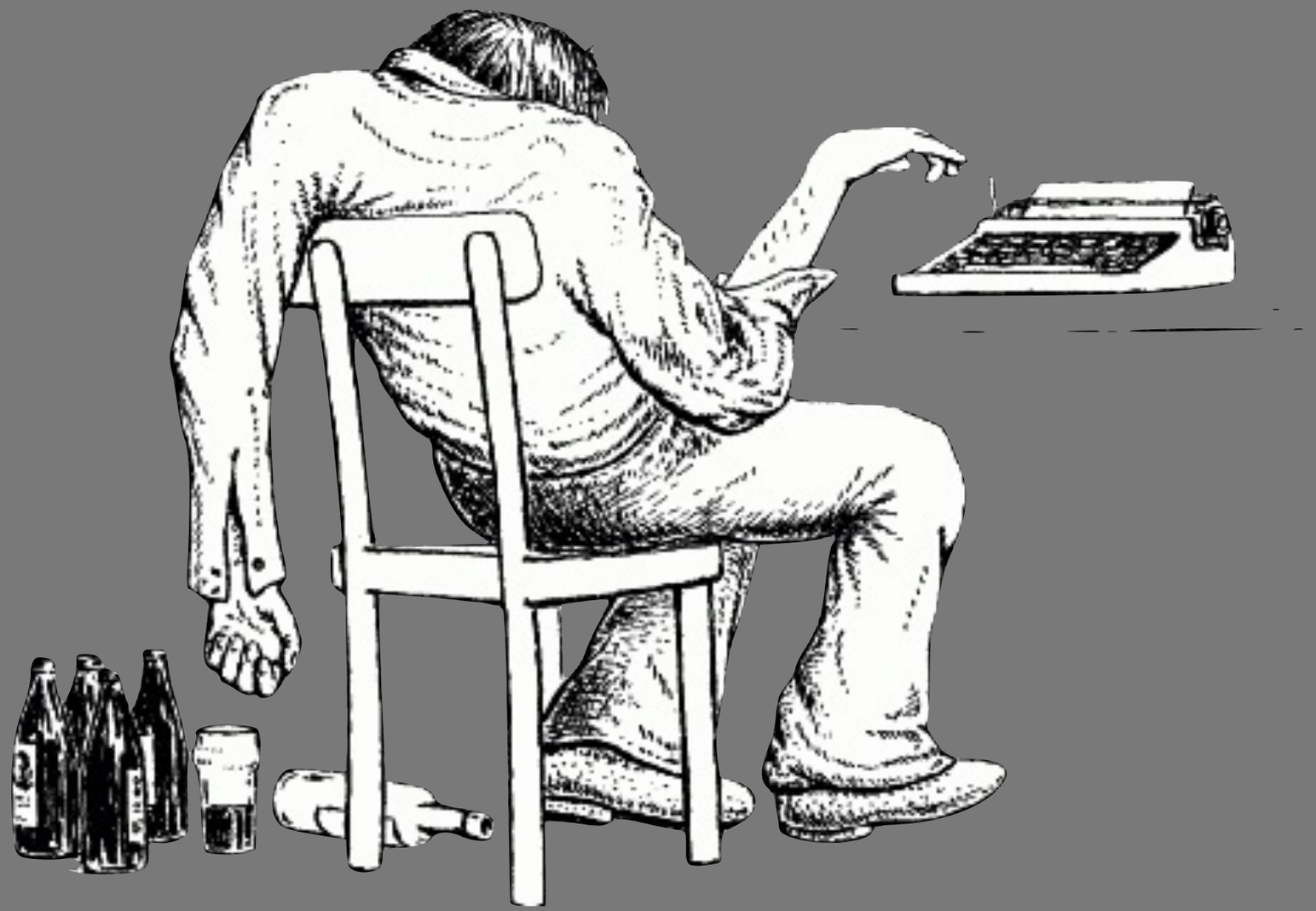
Counted

IPersistentVector

Indexed

Object

IMeta



# Scaffold

```clojure
(defn scaffold
  "Given an interface, returns a 'hollow' body suitable for use with `deftype`."
  [interface]
  (doseq [[iface methods] (->> interface
                               .getMethods
                               (map #(vector (.getName (.getDeclaringClass %))
                                             (symbol (.getName %))
                                             (count (.getParameterTypes %))))
                               (group-by first))]
    (println (str "  " iface))
    (doseq [[_ name argcount] methods]
      (println
       (str "    "
            (list name (into '[this] (take argcount (repeatedly gensym)))))))))
```

‘scaffold’ by Christophe Grand

```
user> (scaffold clojure.lang.IPersistentMap)
  clojure.lang.IPersistentMap
    (assoc [this G__2157 G__2158])
    (without [this G__2159])
    (assocEx [this G__2160 G__2161])
  java.lang.Iterable
    (iterator [this])
  clojure.lang.Associative
    (containsKey [this G__2162])
    (assoc [this G__2163 G__2164])
    (entryAt [this G__2165])
  clojure.lang.IPersistentCollection
    (count [this])
    (cons [this G__2166])
    (empty [this])
    (equiv [this G__2167])
  clojure.lang.Seqable
    (seq [this])
  clojure.lang.ILookup
    (valAt [this G__2168])
    (valAt [this G__2169 G__2170])
  clojure.lang.Counted
    (count [this])
nil
user>
```

Interface we want

Ancestor interface

Method sigs.

# bit.ly/clj-scaffold

steal this code!

# A Note About IFn

```
 1  (invoke [_ a1])
 2  (invoke [_ a1 a2])
 3  (invoke [_ a1 a2 a3])
 4  (invoke [_ a1 a2 a3 a4])
 5  (invoke [_ a1 a2 a3 a4 a5])
 6  (invoke [_ a1 a2 a3 a4 a5 a6])
 7  (invoke [_ a1 a2 a3 a4 a5 a6 a7])
 8  (invoke [_ a1 a2 a3 a4 a5 a6 a7 a8])
 9  (invoke [_ a1 a2 a3 a4 a5 a6 a7 a8 a9])
10  (invoke [_ a1 a2 a3 a4 a5 a6 a7 a8 a9 10])
11  (invoke [_ a1 a2 a3 a4 a5 a6 a7 a8 a9 a10 a11])
12  (invoke [_ a1 a2 a3 a4 a5 a6 a7 a8 a9 a10 a11 a12])
13  (invoke [_ a1 a2 a3 a4 a5 a6 a7 a8 a9 a10 a11 a12 a13])
14  (invoke [_ a1 a2 a3 a4 a5 a6 a7 a8 a9 a10 a11 a12 a13 a14])
15  (invoke [_ a1 a2 a3 a4 a5 a6 a7 a8 a9 a10 a11 a12 a13 a14 a15])
16  (invoke [_ a1 a2 a3 a4 a5 a6 a7 a8 a9 a10 a11 a12 a13 a14 a15 a16])
17  (invoke [_ a1 a2 a3 a4 a5 a6 a7 a8 a9 a10 a11 a12 a13 a14 a15 a16 a17])
18  (invoke [_ a1 a2 a3 a4 a5 a6 a7 a8 a9 a10 a11 a12 a13 a14 a15 a16 a17 a18])
19  (invoke [_ a1 a2 a3 a4 a5 a6 a7 a8 a9 a10 a11 a12 a13 a14 a15 a16 a17 a18 a19])
20  (invoke [_ a1 a2 a3 a4 a5 a6 a7 a8 a9 a10 a11 a12 a13 a14 a15 a16 a17 a18 a19 a20])
21  (invoke [_ a1 a2 a3 a4 a5 a6 a7 a8 a9 a10 a11 a12 a13 a14 a15 a16 a17 a18 a19 a20 & more])
```

Holy arities, Batman!

# A Note About IFn

```
 1  (invoke [_ a1])
 2  (invoke [_ a1 a2])
 3  (invoke [_ a1 a2 a3])
 4  (invoke [_ a1 a2 a3 a4])
 5  (invoke [_ a1 a2 a3 a4 a5])
 6  (invoke [_ a1 a2 a3 a4 a5 a6])
 7  (invoke [_ a1 a2 a3 a4 a5 a6 a7])
 8  (invoke [_ a1 a2 a3 a4 a5 a6 a7 a8])
 9  (invoke [_ a1 a2 a3 a4 a5 a6 a7 a8 a9])
10  (invoke [_ a1 a2 a3 a4 a5 a6 a7 a8 a9 10])
11  (invoke [_ a1 a2 a3 a4 a5 a6 a7 a8 a9 a10 a11])
12  (invoke [_ a1 a2 a3 a4 a5 a6 a7 a8 a9 a10 a11 a12])
13  (invoke [_ a1 a2 a3 a4 a5 a6 a7 a8 a9 a10 a11 a12 a13])
14  (invoke [_ a1 a2 a3 a4 a5 a6 a7 a8 a9 a10 a11 a12 a13 a14])
15  (invoke [_ a1 a2 a3 a4 a5 a6 a7 a8 a9 a10 a11 a12 a13 a14 a15])
16  (invoke [_ a1 a2 a3 a4 a5 a6 a7 a8 a9 a10 a11 a12 a13 a14 a15 a16])
17  (invoke [_ a1 a2 a3 a4 a5 a6 a7 a8 a9 a10 a11 a12 a13 a14 a15 a16 a17])
18  (invoke [_ a1 a2 a3 a4 a5 a6 a7 a8 a9 a10 a11 a12 a13 a14 a15 a16 a17 a18])
19  (invoke [_ a1 a2 a3 a4 a5 a6 a7 a8 a9 a10 a11 a12 a13 a14 a15 a16 a17 a18 a19])
20  (invoke [_ a1 a2 a3 a4 a5 a6 a7 a8 a9 a10 a11 a12 a13 a14 a15 a16 a17 a18 a19 a20])
21  (invoke [_ a1 a2 a3 a4 a5 a6 a7 a8 a9 a10 a11 a12 a13 a14 a15 a16 a17 a18 a19 a20 & more])
```

Should implement all to avoid non-helpful exception.

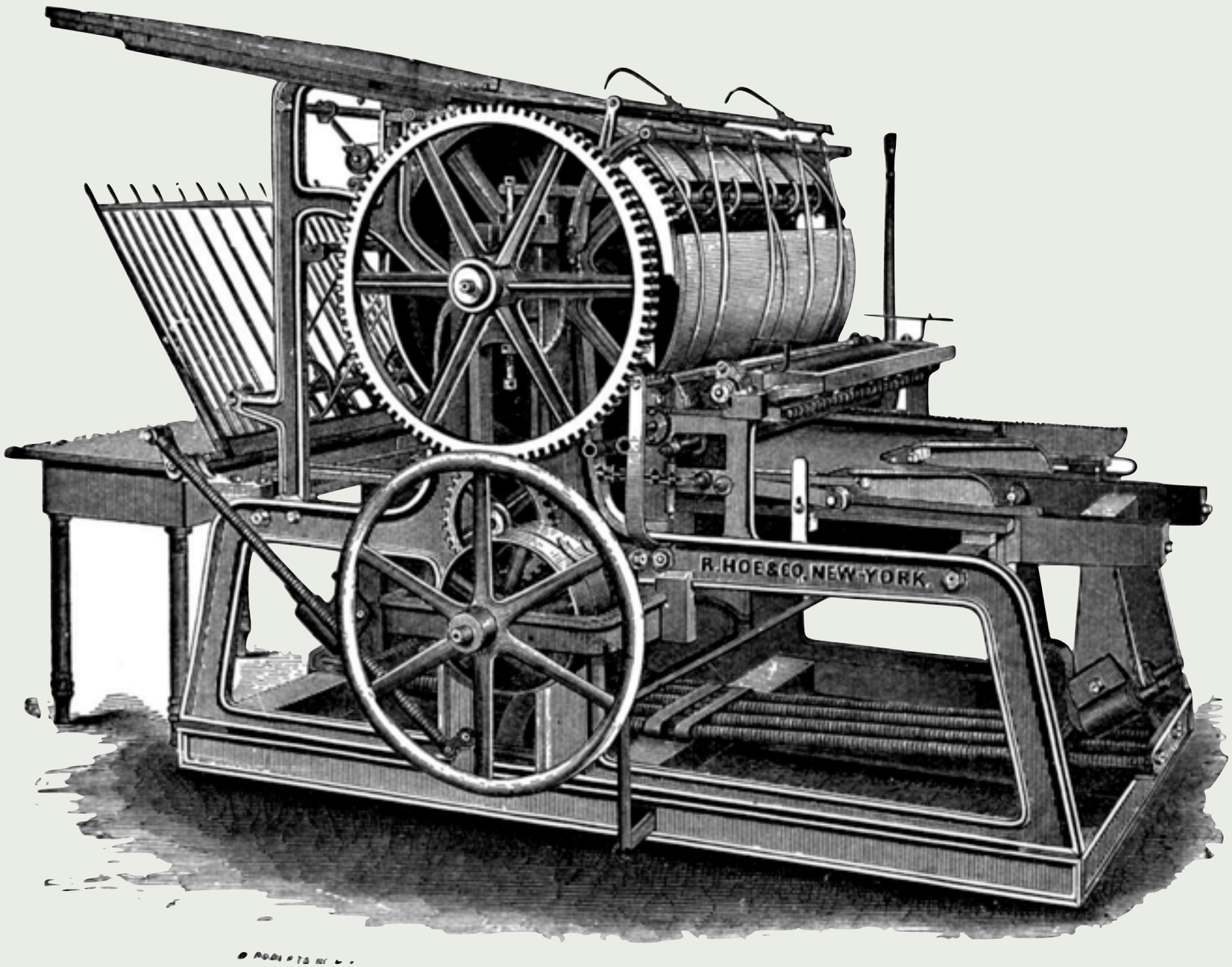
# MUTABLE TYPES

## `^:volatile-mutable`

atomic reads & writes

## `^:unsynchronized-mutable`

a ‘vanilla’ Java mutable field




# Printing

```clojure
(defmethod clojure.core/print-method MyType
  [obj writer]
  (.write writer (str "#MyType<" (things obj) ">")))
```

# clojure.core/print-method

## for humans

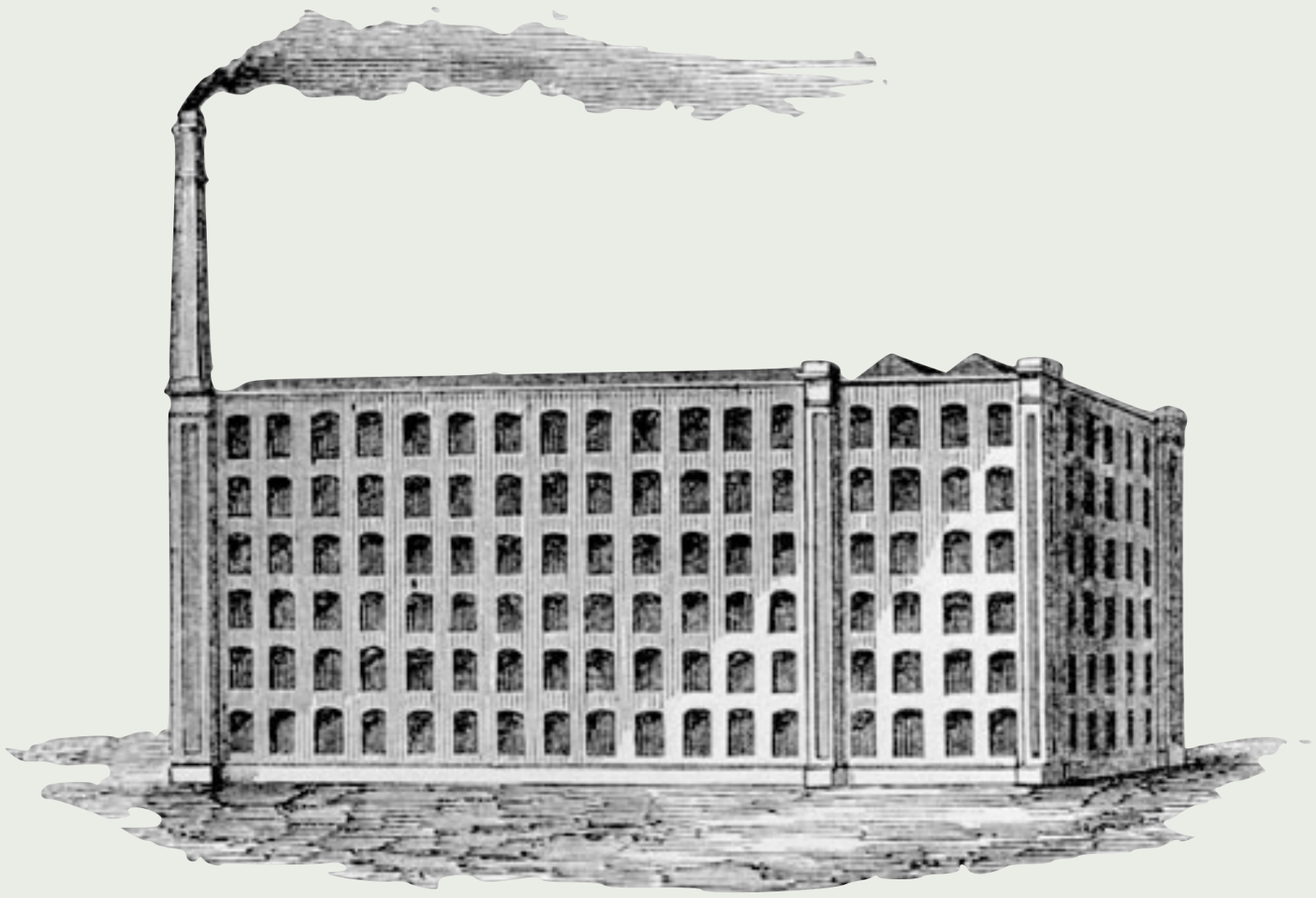
```clojure
(defmethod clojure.core/print-dup MyType
  [obj writer]
  (.write writer (str "#mylib.core.MyType[" (things obj) "]")))
```

# clojure.core/print-dup

## for the reader

# Object Literal Reader Syntax

```
#mylib.core.MyType[1,2,3]
```



# Factory Functions

# Hide Implementation Details

# Avoid Importing Java Classes

```
(my-type :foo :bar)
```

vs.

```
(import myns.MyType)
(MyType. [:foo :bar])
```

# AN EXAMPLE IMPLEMENTATION

(a weekend hack)

# bit.ly/clj-chainmap

steal this code!

*Fin*