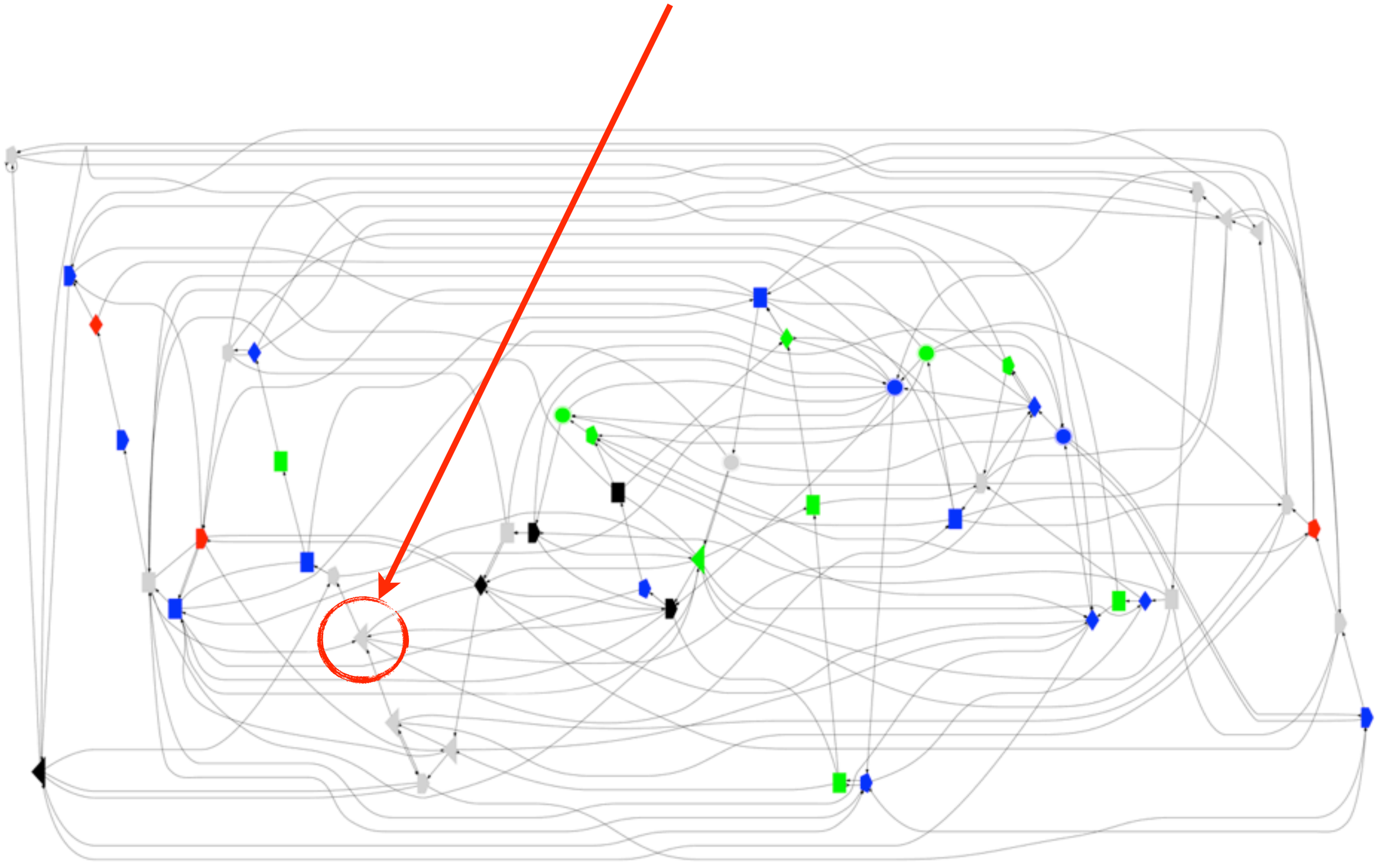
edn and Fressian: Languages of the System

@stuarthalloway

You Are Here



Narcissistic Design

State Transition Model

Processing Model

Topology

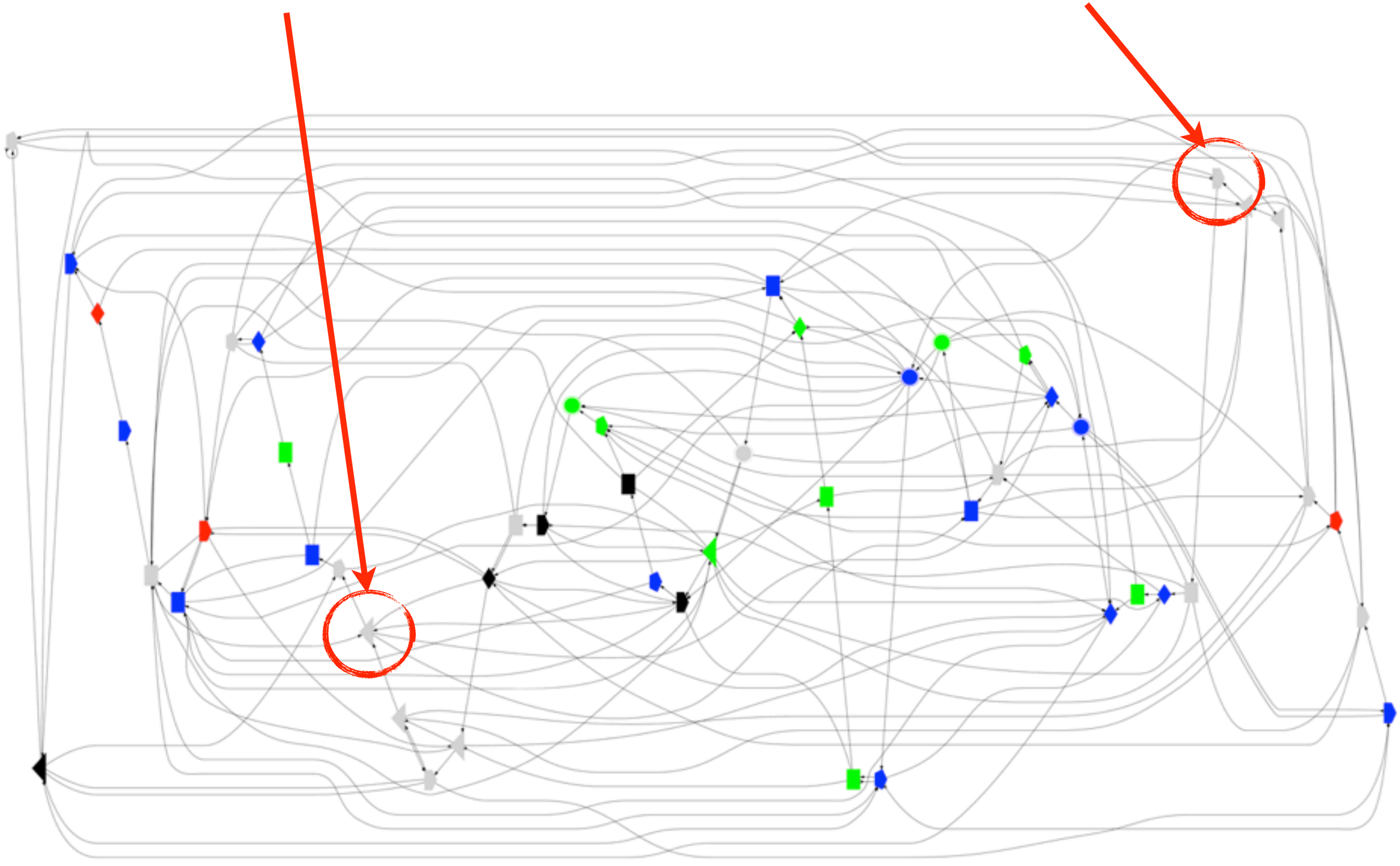
Availability5

Static Typing

Centralized Authority

Language X *(for all values of X!)*

Change Here … Impact Here?



Leverage

Structure

Extensibility

Composability

Universal Data

Self-describing

Immutable Values

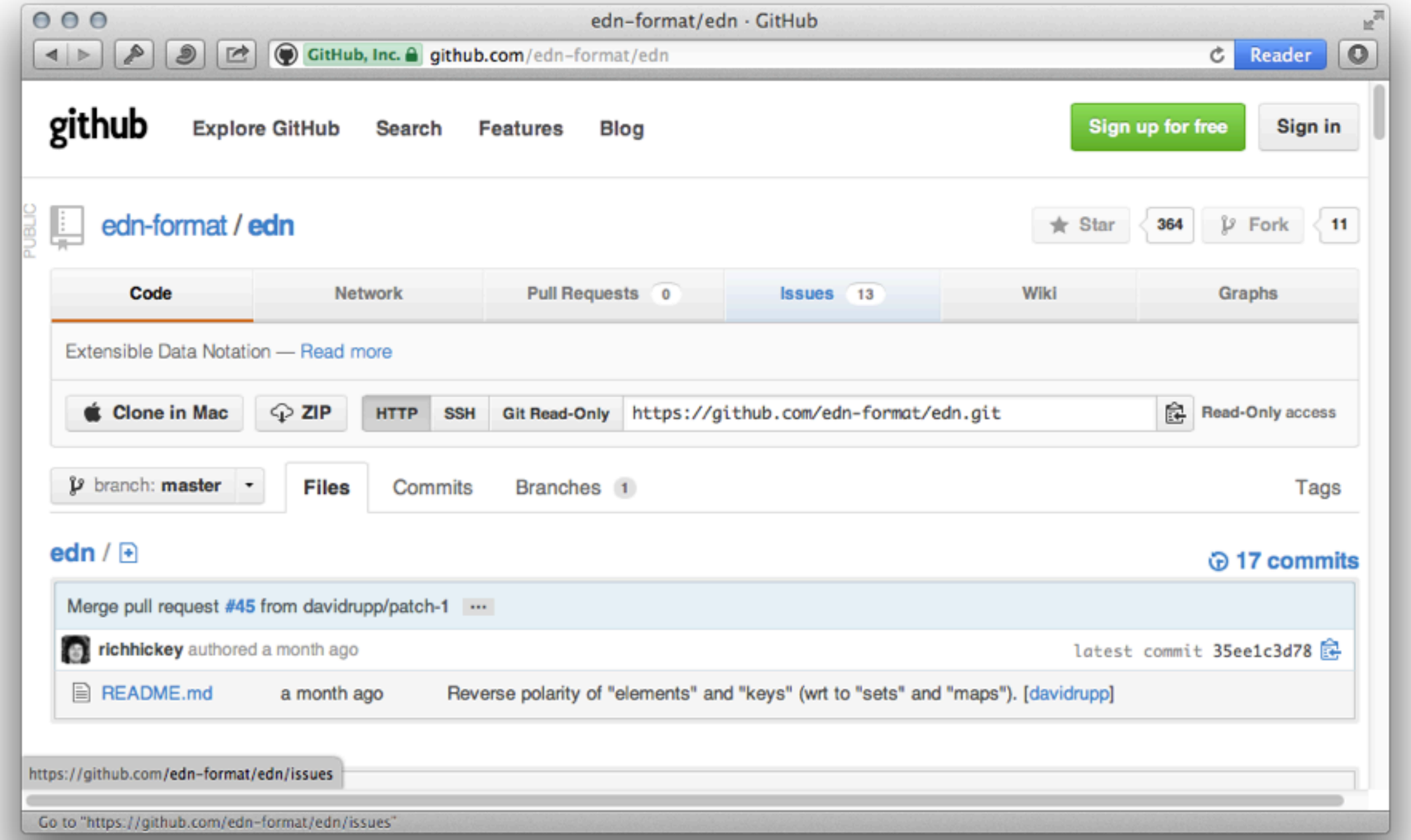
Namespaces

Extensible

Composable

Batteries-included

edn-format.org



edn Example

```
{ :firstName "John"
  :lastName "Smith"
  :age 25
  :address {
    :streetAddress "21 2nd Street"
    :city "New York"
    :state "NY"
    :postalCode "10021" }
  :phoneNumber
    [ {:type "name" :number "212 555-1234"}
      {:type "fax" :number "646 555-4567" } ] }
```

edn is

Subset of Clojure syntax

Used by Datomic and others as data transfer format

Language/implementation neutral

Open

edn Is NOT

Type System

Schema Based

Object-Oriented

Scalars 1

`nil`	nil, null, or nothing
booleans	**`true`** or **`false`**
strings	enclosed in **`"double quotes"`** may span multiple lines **`\t \r \n`** supported
characters	**`\c`** **`\newline`**, **`\return`**, **`\space`** and **`\tab`**

Scalars 2: Numbers

integers	0-9 negative
floating point	64-bit (double) precision is expected.

Scalars 3: Names

symbols	used to represent identifiers should map to something other than strings may include *namespace* prefixs: `my-namespace/foo`
keywords	identifiers that designate themselves semantically akin to enumeration values symbols that must start with **:** `:fred` or `:my/fred`

Collections 1

lists	a sequence of values zero or more elements within **()** **(a b 42)**
vectors	a sequence of values… …that supports random access zero or more elements within **[]** **[a b 42]**

Collections 2

maps	collection of key/value associations every key should appear only once unordered zero or more elements within **{}** **{:a 1, "foo" :bar, [1 2 3] four}**
sets	collection of unique values unordered heterogeneous zero or more elements within **#{}** **#{a b [1 2 3]}**

Disregard

comments	;
discard	#_ is the discard sequence read & discard the next element `[a b #_foo 42] => [a b 42]`

Value Equality

nil, booleans, strings, characters, and symbols are equal to values of the same type with the same edn representation.

integers and floating point numbers should be considered equal to values only of the same magnitude, type, and precision.

sequences (lists and vectors) are equal to other sequences whose count of elements is the same, and for which each corresponding pair of elements (by ordinal) is equal

Value Equality

sets are equal if the have the same count of elements and, for every element in one set, an equal element is in the other.

maps are equal if they have the same number of entries, and for every key/value entry in one map an equal key is present and mapped to an equal value in the other.

tagged elements must define their own equality semantics.

Extensibility: Tagged Elements

```
#name edn-form
```

Name describes interpretation of following element

Recursively defined

Built-in Tagged Elements

`#inst "rfc-3339-format"`

tagged element is a string in RFC-3339 format

`#uuid "f81d4fae-7dec-11d0-a765-00a0c91e6bf6"`

tagged element is a canonical UUID string

Implementing a Tag Handler

associate a qualified name

```
Parser.Config cfg =
    Parsers.newParserConfigBuilder()
    .putTagHandler(Tag.newTag("us.bpsm", "uri"),
        new TagHandler() {
        public Object transform(Tag tag, Object value) {
            return URI.create((String) value);
        }
    }).build();
Parser p = Parsers.newParser(
Parseable pbr = Parsers.newP
    "#us.bpsm/uri \"http://example.com\"");
assertEquals(new URI("http://example.com"), p.nextValue(pbr));
```

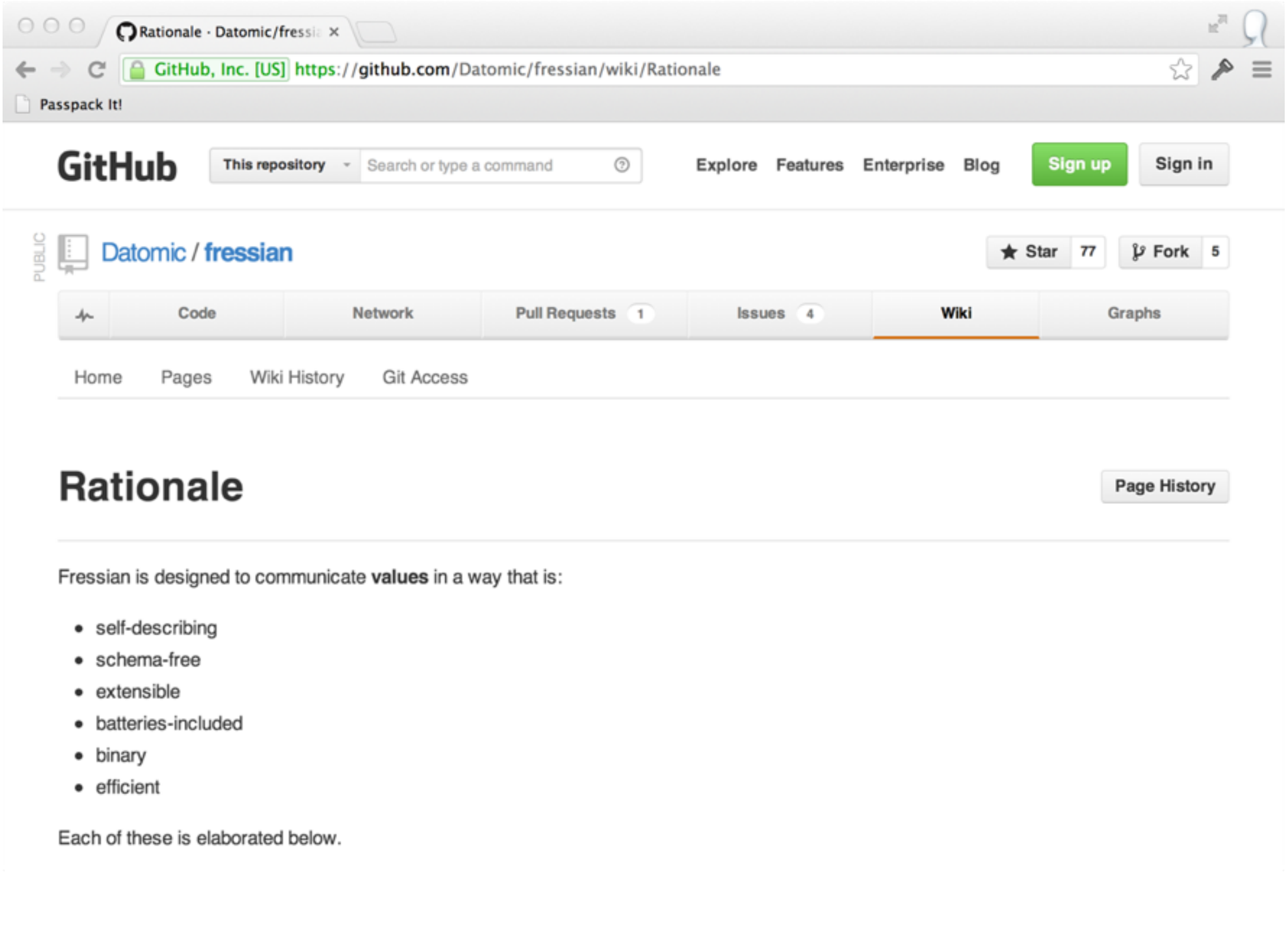
with an interpretation

https://github.com/bpsm/edn-java

Fressian: Lean & Mean

Still no narcissism!

Fressian


Rationale · Datomic/fressian
GitHub, Inc. [US] https://github.com/Datomic/fressian/wiki/Rationale
Passpack It!
GitHub
This repository
Search or type a command
Explore
Features
Enterprise
Blog
Sign up
Sign in
PUBLIC
Datomic / fressian
Star 77
Fork 5
Code
Network
Pull Requests 1
Issues 4
Wiki
Graphs
Home
Pages
Wiki History
Git Access
Rationale
Page History
Fressian is designed to communicate values in a way that is:
self-describing
schema-free
extensible
batteries-included
binary
efficient
Each of these is elaborated below.

Fressian is Similar to edn

Self-describing

Immutable Values

Namespaces

Extensible

Composable

Batteries-included

Fressian is Also

Binary

Byte-code driven

Primitive aware

Domain compressible

Basics

```
public Writer writeObject(Object o);
public Writer writeAs(String tag, Object o);
```

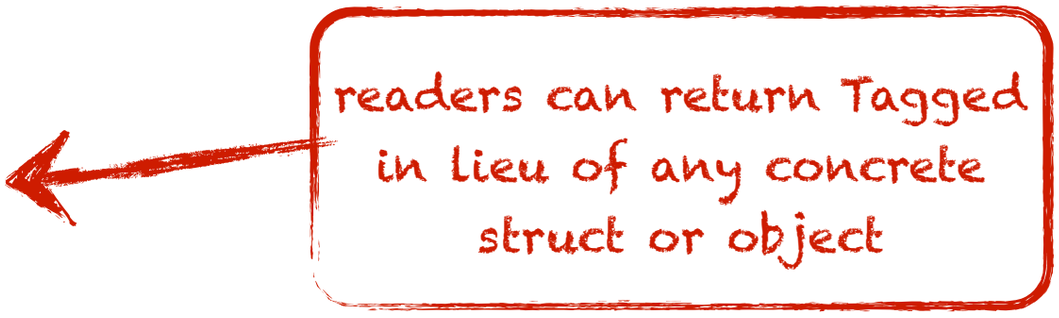
```
public Object readObject();
```

Primitives

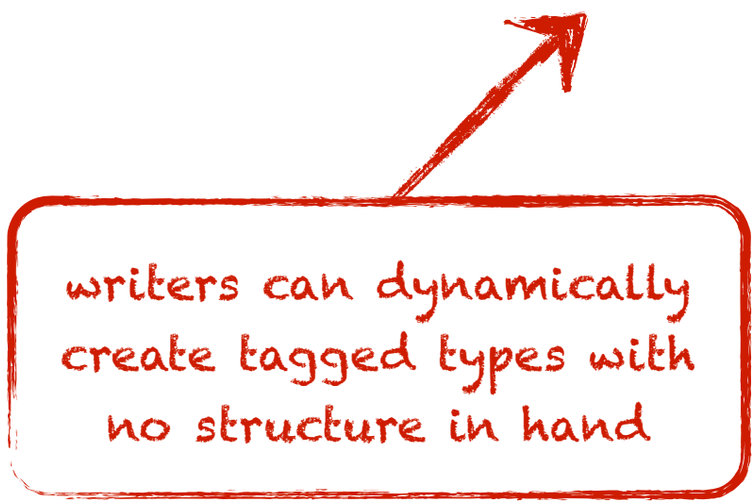
```
public Writer writeBoolean(boolean b) throws IOException;
public Writer writeBoolean(Object o) throws IOException;
public Writer writeInt(long l) throws IOException;
public Writer writeInt(Object o) throws IOException;
public Writer writeDouble(double d) throws IOException;
public Writer writeDouble(Object o) throws IOException;
public Writer writeFloat(float d) throws IOException;
public Writer writeFloat(Object o) throws IOException;
public Writer writeString(Object o) throws IOException;
public Writer writeList(Object l) throws IOException;
public Writer writeBytes(byte[] b) throws IOException;
public Writer writeBytes(byte[] b, int offset, int length);
```

Understanding is Optional

```
public interface Tagged {
      public Object getTag();
      public Object getValue();
      public Map getMeta();
}
```




```
public Writer writeTag(Object tag, int componentCount);
```



Caching

tell writer to cache...

```
public Writer writeObject(Object o, boolean cache);
public Writer writeAs(String tag, Object o, boolean cache);
```

```
public interface Cached {
    public Object getObjectToCache();
}
```

...or object can know it wants caching

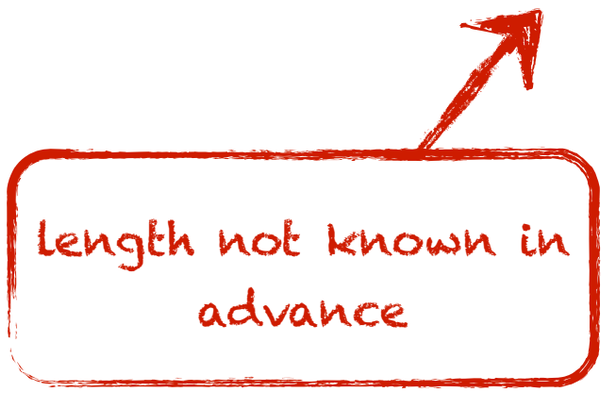
Caching Example

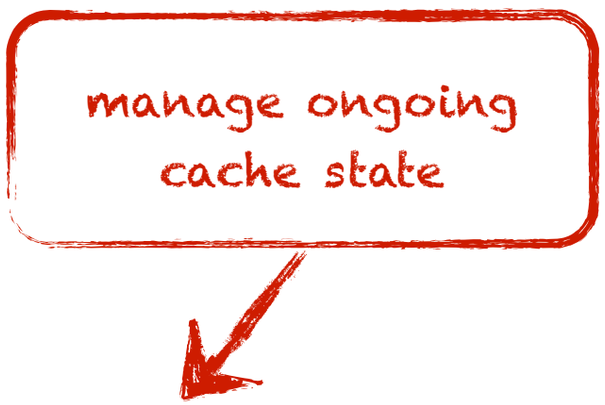
```
new WriteHandler() {
    public void write(Writer w, Object instance) throws IOException {
        w.writeTag(tag, 6);
        Datom d = (Datom) instance;
        // elided write five other tags
        w.writeObject(d.getT(), true);
    }
};
```

Datomic efficiently stores the time T of every atomic fact in the system

Streaming

```
public Writer beginClosedList() throws IOException;
public Writer endList() throws IOException;
```






```
writer.resetCaches();
```

Packed Representations

Encode type and part|all of a value in a single byte

`0x00` means “numeric” and “value 0”

`0xd1` mean “byte array” and “length 1”

Chunked Representations

Work in constrained memory, nothing > 64k

`0xE2` precedes nonterminal chunk of a string

`0xE3` preceded terminal chunk of a string

Similar for bytes

Byte Code Overview

0 - 0x40	*themselves*
0x40 - 0x80	*cascading packed integers*
0xF8	*a.p. integer*
0xD0 - 0xD8	*packed byte arrays*
0x80 - 0xA0	*single-byte cache codes*
0xCC, 0xCD	*get/put cache*
0xC5	*URI*

https://github.com/Datomic/fressian/wiki/Bytecodes

https://github.com/Datomic/fressian/blob/master/test/org/fressian/codegen.clj

edn and Fressian are

Value formats

Self-describing

Extensible

Schemaless

Designed for an decentralized world

Languages of the System

Domain-Specific Data Languages (DSDLs)

Sample Dataset

entity	attribute	value
42	:email	John
42	:orders	77
77	:items	101
101	:price	10.99
77	:items	103
103	:price	11.77

How much money has John spent in the store?

Query Language Design

People build queries

Programs build queries

Store and transmit queries

DSL

```
SELECT customer, price
(FROM universe)
WHERE customer = 42
customer.order = order
order.lineitem = li
li.price = price
```

API

```
Query q = new Query();
q.addConstraint(new Variable("?customer"),
                new Constant("orders"),
                new Variable("?order"));
q.addConstraint(new Variable("?order"),
                new Constant("item"),
                new Variable("?item"));
q.addConstraint(new Variable("?item"),
                new Constant("item"),
                new Variable("?price"));
q.setVariable("?customer", 42);
q.find("?customer", "?price");
q.setDatabase(db);
Result Set rs = q.execute();
```

Bottom Types -> Data

```
Query q = new Query();
q.addConstraint("?customer", "orders", "?order");
q.addConstraint("?order", "item", "?item");
q.addConstraint("?item", "item","?price"));
q.setVariable("?customer", 42);
q.find("?customer", "?price");
q.setDatabase(db);
Result Set rs = q.execute();
```

strings instead of Constant and Variable classes

Introduce Name Types

```
Query q = new Query();
q.addConstraint(?customer, :orders, ?order);
q.addConstraint(?order, :item, ?item);
q.addConstraint(?item, :item, ?price));
q.setVariable(?customer, 42);
q.find(?customer, ?price);
q.setDatabase(db);
Result Set rs = q.execute();
```

symbols for variables and
keywords for attribute names

Mutable Objects -> Immutable Collections

```
where = [[?customer :orders ?order]
         [?order :item ?item]
         [?item :price ?price]]
inputs = {?customer 42}
find = [?customer ?price]

query(database, where, inputs, find);
```

Entire Query as Data

```
query(database,
      {:where [[?customer :orders ?order]
               [?order :item ?item]
               [?item :price ?price]]
       :inputs {?customer 42}
       :find [?customer ?price]});
```

DSL / API / DSDL

	DSL	**API**	**DSDL**
human use	friendly	hostile	friendly
programmatic use	hostile	custom API	generic API
serialization	hunh?	hunh?	free
concurrency	hunh?	hunh?	free

Resources

The Language of the System
https://www.youtube.com/watch?v=ROor6_NGIWU

edn
http://edn-format.org. The edn specification.
http://clojure.com. The Clojure language.
https://github.com/edn-format/edn/wiki/Implementations. edn implementations,

Fressian
https://github.com/Datomic/fressian. The Fressian spec and reference impl.
http://www.datomic.com/. Datomic.
https://github.com/fressian/fressian-clr. Fressian on the CLR.

Stuart Halloway
https://github.com/stuarthalloway/presentations/wiki. Presentations
http://thinkrelevance.com/blog/tags/podcast. The Relevance Podcast.
http://www.linkedin.com/pub/stu-halloway/0/110/543/
https://twitter.com/stuarthalloway