U5ZAJ15P0 : thanks!
U0BKWMG5B : Right
U0BKWMG5B : Suspend and resume are useful for keeping connections open and avoiding restart time during development.
U0BKWMG5B : Web sockets for example
U628K7XGQ : How are you guys printing the guts of Java objects on the REPL? I'd love to be able to convert any java object to a graph of maps I can easily inspect.(similar to what the debugger in IntelliJ allows you to do)

U060FKQPN : `bean`
U628K7XGQ : wouldn't work on non-beans
U628K7XGQ : I tried to make sense of `clojure.reflect` but before I spent my time on that - shall we say - sparsely documented lib, I was wondering if someone had a neat `java-to-map` fn
U050MP39D : I think clojure.reflect/reflect basically does that iirc
U051SS2EU : if I recall, bean used to be more general and didn't look for JavaBean specific stuff... let me see if I can find it
U051SS2EU : ```+user=&gt; (bean (java.util.Date.)){:day 6, :date 22, :time 1500741217657, :month 6, :seconds 37, :year 117, :class java.util.Date, :timezoneOffset 420, :hours 9, :minutes 33}
```

U051SS2EU : maybe it's that Date is a bean and I never realized it was?
U050MP39D : yeah bean looks quite good```
(bean (HTableDescriptor. (TableName/valueOf "foo")))
=&gt;
{:familiesKeys #{},
 :columnFamilies #object["[Lorg.apache.hadoop.hbase.HColumnDescriptor;"
                         0x124eda1b
                         "[Lorg.apache.hadoop.hbase.HColumnDescriptor;@124eda1b"],
 :regionReplication 1,
 :memStoreFlushSize -1,
 :name #object["[B" 0x19dca84c "[B@19dca84c"],
 :tableName #object[org.apache.hadoop.hbase.TableName 0x6328ce6a "foo"],
 :metaRegion false,
 :rootRegion false,
 :compactionEnabled true,
 :maxFileSi... snip
```

U051SS2EU : and you could start with bean's source code if you want something that isn't so JavaBean centric
U065JNAN8 : I don't understand this line in the docs about clojure.spec
"Thus a bare (s/keys) is valid and will check all attributes of a map without checking which keys are required or optional."

What is it checking if there are no specs to check?

```
(s/def ::foo (s/keys))
(s/valid? ::foo {:yolo 42})
=&gt;true
```

U065JNAN8 : Ah I hadn't read enough. Namespace qualified keys will be checked against their specs
U050R7ECY : Are there any good debuggers that work at the bytecode level? I have cider, but I have an infinite loop somewhere, so I need 'pause execution'
U051SS2EU : cider and cursive have this - to some degree, you need to turn off some clojure features for it to work well
U051SS2EU : usually I opt for adding (swap! debug-atom conj {:context ::foo :data foo}) in the middle of some function and then use the resulting atom in the repl to figure out what's going on (which does require iterating sometimes to figure out which data you should even be tracking of course)
U5ZAJ15P0 : <@U051SS2EU> is there an equivalent of Ruby's "pry" in clojure? e.g. a way to open a repl at any point in a program, with access to local bindings?
U050R7ECY : I thought cider's debugger worked by instrumenting and evaling source?

U051SS2EU : <@U050R7ECY> I didn't check recently but at one point they were actually using jdx bindings to get into the byte code level - when I saw people claim "cider can debug like cursive now" I assumed that meant that feature was working