U051SS2EU : right- no evidence of contention there at all
U06BV1HCH : fwiw the alternative we're considering would be function args, rather than vars, with the config passed everywhere
U06BV1HCH : awesome, thx!!
U051SS2EU : I realize I didn't benchmark just having a data literal - checking that now to see if there's any difference worth noticing
U051SS2EU : but really profiling your own code and looking for the actual perf bottlenecks is the way to go
U051SS2EU : yeah - replacing @a with a literal 42 (but still reducing over futures etc.) ends up taking ~the same~ (actually slightly longer, but within the measurement epsilon so not meaningful)
U06BV1HCH : great to know -- thanks!
U5ZAJ15P0 : Hello! I am using Pedestal + Lacinia Pedestal to develop a GraphQL server, but I am not quite sure how to set up hot-reloading. Could anyone point me in the right direction?
U5ZAJ15P0 : I would like to hot-reload the GraphQL schema and the resolvers
U0514TE0F : Hi, could you point me some http/2/streaming clojure client?
U640YAFPW : We are trying to add clojure.spec and metosin/spec-tools to our codebase, and it looks like clojure 1.9 is a prerequisite. We tried upgrading to `1.9-alpha17`, but currently facing a bunch of problems related to the upgrade. Our app currently does not start, with errors similar to the followingZ```Caused by: clojure.lang.ExceptionInfo: Call to clojure.core/defn- did not conform to spec:
In: [0] val: clj-tuple/conj-tuple fails spec: :clojure.core.specs.alpha/defn-args at: [:args :name] predicate: simple-symbol?```

A number of these are in libs and stuff. We've been upgrading a few of them, or fixing the error, which typically causes a new error to appear. Is there anyway to switch off clojure.spec validating the entire codebase + libs? Possibly get it to stick to a few namespaces only? Is this happening because we are trying to upgrade to an alpha?

U060FKQPN : there's no way to turn off spec checks on macroexpansion
U060FKQPN : your only option is to upgrade to versions of the libraries that include the fixes you need
U050DD55V : hi - is there a way to get all resources on the classpath - whether they be inside of jars of on the filesystem?
U0FR9C8RZ : <@U050DD55V> I'm sure there must be but I don't know what it is. eg. iirc Spring does a bunch of classpath scanning
U06BE1L6T : jonpither: something like this (in java): <https://stackoverflow.com/questions/3923129/get-a-list-of-resources-from-classpath-directory>
U640YAFPW : <@U060FKQPN> thanks
U1WAUKQ3E : Hi! I wrote an article about PostgreSQL to Datomic migration. Hope you'll find it useful.<http://grishaev.me/en/pg-to-datomic>

U5ZAJ15P0 : igrishaev: thank you for writing this up; very useful!
U5ZAJ15P0 : there is a <#C03RZMDSH|datomic> channel, you might want to post it there too
U1WAUKQ3E : thank you, makes sense
U0CKDHF4L : does the describe of a spec always have a similar structure to the data it specifies ? Can you give a counterexample ?
U5ZAJ15P0 : <@U0CKDHF4L> I am not overly familiar with clojure specs, but map specs are represented as vectors I think
U5ZAJ15P0 : well, as a map with a `:req` vector
U0CKDHF4L : yes, ```(keys :req etc)``` -- but that has basically the same nesting structure
U0CKDHF4L : I'm trying to think of a case where you describe a nested structure of collections by a flat spec
U0GC1C09L : does clojure have a representation of infinity that can work with mathematical operators? something like `(&lt;= 1 2 Infinity)` ?
U0CKDHF4L : ```Double/POSITIVE_INFINITY```
U0CKDHF4L : or ```Number.POSITIVE_INFINITY``` in CLJS
U0GC1C09L : !! thanks :slightly_smiling_face:
U1NGX4Z6F : hey guys
U1NGX4Z6F : what's you're preferred method of checking for nils before assignation ?
U060FKQPN : <@U0GC1C09L> see also <https://dev.clojure.org/jira/browse/CLJ-1074>
U1NGX4Z6F : i just realized I am repeating `(:username respone)` when I `(if-not (nil? (:username response) (:username response) fallback-value))`