



(f/insert-file on {*name*(getNamel file):*prosed*(Date.)})

```
(defn file-handler [{:keys [queue]} {:keys [^File kind] :as event}]
```

```
(defn weapon-query-handler [{:status [name]} params :keys [conn] :as request}]
```

```
(defn add-weapon-handler [{:keys [params] :as request}]
```

{:jdbc/connnection{:connnection-uniq'jdbc:h2:men:men\_only'}}

(timbre/debug(str 'Detected hang to file: ' (getNamel file)))

```
(let [name & weapons] (map cs/trim (cs/spplit line #', ''))])
```



*;;jdbc/init {;conn (ig/ref::jdbc/conn)}*

```
(let [data (map(fn [[kv]] {:name weapon v}) params)]
```

*; (timbre/debug'Checking for new items in queue.')*

(defmethod dig/init-key::jdbc/init[\_ {::keys [conn]}]

`awk/watch { :groups { :paths ['example']`

*is checking job* { *job* #'queue->dsdb

```
(defn queue->dsdb [{:keys [queue-name queue-dsdb]}])
```

`(when-some [task (dq/take! queue-name 10)])`



`{:data{:precisionreit.it.cerion.spec/cerion`

*if (isAbLe/queues { isDet-e-n-halt? true*

(tinbre/debug 'Putting data into data script')

*is cdul { : in [5 : seconds : every : second }*

*:dsdb(ig/ref::data\$ip\$connection)*

#!/web/server { :host "0.0.0.0"

*:con (ig/ref::data input/connetion)*

(d//transact!dsdb[(line->read@task)])



`(ok(without-str(pp/print request)))`

```
(defn filter-handler [{:keys [sql-conn]}])
```

*We were able to connect the routes here*

**(ok(w/weapons@acorn(keysparams)))**

(cs/ends-with? (.getNamel file) ".csv"))

**(ok(f/all-proposed-files ql-con))**

(constantly(not-found'Not found'))

(tinbre/debug 'Adding data to queue.')



['/add\_webon' add-weapons-handler])

*:queue (ig/ref::durab le/queue)*

(when (and (#*modify* *read*) kind)

*sqz-conn(ig/ref::jdb/c/connec<sup>ti</sup>on)*

['/weapons'-query-handler]

*{conn(i:ig/ref::j:dbc/connec tion)}*

```
["/files"{:agetfiles-handler}] ] )
```

*:queue (ig/ref:duration/queue)*



(dq//put!queue:my-queue line)

*data/cnript/cnnnet\_idn* w/schema

(w/everybody's weapons @onn))

*mind/leware* [params/wrap-params]

(with-open [r(io//reader file)])

```
[[ '/echo' { :get echo-handler } ]]
```

[baisic-routes-weapons-routes]

(define sys(create conf))



(defn echo-handler [request])

mind[eware//wrap-format]})

(assoc: *wrap* *wrap*)

(doseq[line](line-seq r))]

(define-line->record [line])

*hand/zer* # *final-hand/zer*]

(w/weapons@corn[ame])

`:'directory'"/tmp'}`



(d/transact!commanddata)

**(dq/complete!task)()**

*#hand/ler* #'hand/ler}})

queque-nammy-queque

(cond=>{:name name})

(def weapons-routes

*Routes - All data*

**(ring/ring-handler**



(defbasic-routes

Agglutiner

**(f/setupcon)**

**(.exits file)**

**(.isFile file file)**

*print* 30000

(seq weapons)

**(ring/router**



def handler

defrouter

(defconfid)

*Handzlers*

)

i

f

n

a

m

e

*Router*

roster

ctx)



(

o

k





