


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

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

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

{:jdbc/connection{:connection-unix'jdbc:h2:mem:mem_only'}}

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

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

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



```

::jdbc/init::jdbc/connnection}

```

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

`hawk/watch { :groups { :paths ["example"] }`

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

;(tindre/debug'checking for new items in queue.')

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

(when-one [task (dq/take! queue-name 10 nil)])

is cduling/job {*:job* #'queue->dsdb

`{:data{:precisionreit.it.cerec/cerec`

`is_durable_queue { is_delete-on-halt? true`

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

is chdule { : in [5 : seconds : every : second }

:dsdb(ig/ref::datasrcip/connnection)}

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

:cn (ig/ref::data\$cnip/cnnet)

(d//t rand t! d sdb[(line->read@task)])

(ok(withoutstr(pp/printrequest)))

;We were able to connect the routes here

```
(defn files-handler [{:keys [sql-con]}])
```

(ok(w/weapons@anon(keyparams)))

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

(ok(fatal-procesed-files ql-con))

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

['/add_weapon' add_weapon_handler])

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

:queue (ig/ref::durableref/queue)

sqz-conn(ig/ref::jdb/c/connec tion)

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

[['/weapons' weapons_query_handler]]

{conn(i_g/ref::j_dbc/connec tion)}

```
["/files":get_files_handler]])
```

:queue (ig/ref:duration/queue)

*data/crypt/connections/*schema

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

(w/everybody's-weapon@com))

minddleware [params/wrap-params]

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

[[['/echo'},{:get_echo_handler}]]

[basic-route weapons-route]

(define-sys(create-confing))

(defnetcher [request])

mind[eware//wrap-format]})

(as doc: *weapon* weapons)

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

(define-line->record [line])

hand/ler # 'final-*hand*ler']

(w/weapons@corn[ame])

`:"directory"//tmp"}`

(d/transact!commanddata)

`(dq//complete!task)()`

#hand/ler #'handler})

queue-name:my-queue

(cond->{*name* name})

(def weapons-routes

Routes - Atlanta

(ring/ring-handler

;Gzobazhander

(defbasic-inputs

$(f \setminus \text{setup} \text{ on } n)$

(.isFile file file)

(.exits file)

(ring/router

(seq weapons)

print:30000

Handletter

defrouter

(defconfio)

Handzlers

)

i

f

n

a

m

e

Router

rowt er

ctx)

(

O

K

(

do

o





