Clojure-listing i pandoc

Truls Thirud

Eksempelkode

Her kommer en kodeliste med clojure-kode som bør være fargekodet og i en liten font.

```
(ns guestbook.core
  (:require
    [guestbook.handler :as handler]
    [guestbook.nrepl :as nrepl]
    [luminus.http-server :as http]
    [luminus-migrations.core :as migrations]
    [guestbook.config :refer [env]]
    [clojure.tools.cli :refer [parse-opts]]
    [clojure.tools.logging :as log]
    [mount.core :as mount])
  (:gen-class))

;; log uncaught exceptions in threads
(Thread/setDefaultUncaughtExceptionHandler
  (reify Thread$UncaughtExceptionHandler
    (uncaughtException [_ thread ex]
      (log/error {:what :uncaught-exception
                  :exception ex
                  :where (str "Uncaught exception on" (.getName thread))}))))

(def cli-options
  [["-p" "--port PORT" "Port number"
    :parse-fn #(Integer/parseInt %)]])

(mount/defstate ^{:on-reload :noop} http-server
  :start
  (http/start
    (-> env
        (assoc  :handler (handler/app))
        (update :port #(or (-> env :options :port) %))
        (select-keys [:handler :host :port :async?])))
  :stop
  (http/stop http-server))

(mount/defstate ^{:on-reload :noop} repl-server
  :start
  (when (env :nrepl-port)
    (nrepl/start {:bind (env :nrepl-bind)
                  :port (env :nrepl-port)}))
  :stop
  (when repl-server
    (nrepl/stop repl-server)))


(defn stop-app []
  (doseq [component (:stopped (mount/stop))]
    (log/info component "stopped"))
  (shutdown-agents))

(defn start-app [args]
  (doseq [component (-> args
                        (parse-opts cli-options)
                        mount/start-with-args
                        :started)]
    (log/info component "started"))
  (.addShutdownHook (Runtime/getRuntime) (Thread. stop-app)))

(defn -main [& args]
  (-> args
      (parse-opts cli-options)
      (mount/start-with-args #'guestbook.config/env))
  (cond
    (nil? (:database-url env))
    (do
      (log/error "Database configuration not found, :database-url environment variable must be set before running")
      (System/exit 1))
    (some #{"init"} args)
    (do
      (migrations/init (select-keys env [:database-url :init-script]))
      (System/exit 0))
    (migrations/migration? args)
    (do
      (migrations/migrate args (select-keys env [:database-url]))
      (System/exit 0))
    :else
    (start-app args)))
```