디스패치 다시 생각하기

클로저로 만든 학점 프로그램

```
(ns lettergrades)

(defn in [score low high]
  (and (number? score) (<= low score high)))
(defn letter-grade [score]
  (cond
    (in score 90 100) "A"
    (in score 80 90) "B"
    (in score 70 80) "C"
    (in score 60 70) "D"
    (in score 0 60) "F"
    (re-find #"[ABCDFabcdf]" score) (.toUpperCase score)))
```

클로저 학점 프로그램 테스트

```
(ns lettergradestest
  (:use clojure.test)
  (:use lettergrades))
(deftest numeric-letter-grades
  (dorun (map #(is (= "A" (letter-grade %))) (range 90 100)))
  (dorun (map #(is (= "B" (letter-grade %))) (range 80 89)))
  (dorun (map #(is (= "C" (letter-grade %))) (range 70 79)))
  (dorun (map #(is (= "D" (letter-grade %))) (range 60 69)))
  (dorun (map #(is (= "F" (letter-grade %))) (range 0 59))))
(deftest string-letter-grades
  (dorun (map #(is (= (.toUpperCase %)
                      (letter-grade %))) ["A" "B" "C" "D" "F" "a" "b" "c"
"d" "f"])))
(run-all-tests)
```

클로저로 색 구조를 정의하기

```
(defstruct color :red :green :blue)

(defn red [v]
  (struct color v 0 0))
(defn green [v]
  (struct color 0 v 0))
(defn blue [v]
```

```
  (struct color 0 0 v))
```

멀티메서드 정의하기

```
(defn basic-colors-in [color]
  (for [[k v] color :when (not= v 0)] k))
(defmulti color-string basic-colors-in)

(defmethod color-string [:red] [color]
  (str "Red: " (:red color)))
(defmethod color-string [:green] [color]
  (str "Green: " (:green color)))

(defmethod color-string [:blue] [color]
  (str "Blue: " (:blue color)))
(defmethod color-string :default [color]
  (str "Red: " (:red color) ", Green: " (:green color ", Blue: " (:blue
color))))
```

관련 문서

Plugin Backlinks: 아무 것도 없습니다.