GO QUICK REFERENCE GUIDE net/http

http.HandlerFunc

```go
// Standard HTTP handler function signature.
func myHandler(w http.ResponseWriter, r *http.Request) {
    ...
}

// Standard middleware function signature.
func myMiddleware(next http.Handler) http.Handler {
    return http.HandlerFunc(func(w http.ResponseWriter, r *http.Request) {
        ...
        next.ServeHTTP(w, r)
    })
}
```

http.Request

```go
ip := r.RemoteAddr // Get the IP address that sent the request.
method := r.Method // Get the request method.
path := r.URL.Path // Get the request URL path.

id := r.URL.Query().Get("id")    // Get the query string parameter "id".
accept := r.Header.Get("Accept") // Get the request "Accept" header.

body, err := ioutil.ReadAll(r.Body) // Read the request body into a []byte slice.
```

http.ResponseWriter

```go
// Remember: All headers must be set before calling w.WriteHeader() or w.Write().
w.Header().Set("Location", "/article/1")
w.WriteHeader(201)
w.Write([]byte("Created"))
```

http.NotFound and http.Error

```go
http.NotFound(w, r)
http.Error(w, "Bad Request", 400) // Or using constants...
http.Error(w, http.StatusText(http.StatusBadRequest), http.StatusBadRequest)
```

http.ServeMux

```go
mux := http.NewServeMux()
mux.HandleFunc("/foo", fooHandler)  // Matches the URL path "/foo" only.
mux.HandleFunc("/bar/", barHandler) // Matches all URL paths that begin "/bar/".

// Register a static file server for all URL paths that begin "/static/".
fs := http.FileServer(http.Dir("./path/to/static/dir"))
mux.Handle("/static/", http.StripPrefix("/static", fs))
```

http.Server

```go
srv := &http.Server{
    Addr:         ":4080",
    Handler:      mux,
    ErrorLog:     log.New(...),
    IdleTimeout:  time.Minute,
    ReadTimeout:  5 * time.Second,
    WriteTimeout: 10 * time.Second,
}

err := srv.ListenAndServe()
log.Fatal(err)
```

net/http Constants

```go
http.MethodGet  // "GET"
http.MethodPost // "POST"

http.StatusMovedPermanently    // 301
http.StatusSeeOther            // 303
http.StatusTemporaryRedirect   // 307
http.StatusBadRequest          // 400
http.StatusUnauthorized        // 401
http.StatusForbidden           // 403
http.StatusMethodNotAllowed    // 405
http.StatusInternalServerError // 500
```

http.Request.Form

```go
err := r.ParseForm()       // Must be called before using r.Form or r.PostForm.
id := r.PostForm.Get("id") // Get "id" value from POST or PUT body.
id := r.Form.Get("id")     // Get "id" value from POST or PUT body OR query string.
```

http.Request.Context

```go
type contextKey string
var contextKeyEmail = contextKey("email")

// Add a value to request context. Note: This creates a new copy of the request.
ctx = context.WithValue(r.Context(), contextKeyEmail, "me@example.com")
r = r.WithContext(ctx)

// Retrieve the value from request context.
email, ok := r.Context().Value(contextKeyEmail).(string)
```