Trabajo Práctico 1 - Programación Web

Ejercicio 1: Servidor HTTP Básico en Go

Objetivo: Implementar un servidor HTTP que responda con HTML básico.

1. Crea un archivo `main.go` con un servidor que:
   - Escuche en el puerto 8080
   - Responda a GET / con:
     ```
     <!DOCTYPE html>
     <html>
     <head><title>Hola Mundo</title></head>
     <body><h1>¡Servidor Funcionando!</h1></body>
     </html>
     ```
   - Establezca correctamente el header `Content-Type`
2. Extiende el servidor para:
   - Agregar una ruta `/about` que muestre información sobre el servidor
   - Manejar correctamente rutas no existentes (404)

Ejercicio 2: Formularios y Métodos HTTP

Objetivo: Practicar el manejo de formularios HTML y métodos HTTP.

1. Crea un formulario HTML que:
   - Tenga campos para: nombre, email y mensaje
   - Use método POST
   - Envíe los datos a `/contacto`
2. Implementa el handler para:
   - Mostrar el formulario en GET `/contacto`
   - Procesar los datos en POST `/contacto`
   - Validar que los campos no estén vacíos
   - Mostrar los datos recibidos en una página de confirmación
3. Modifica el formulario para:
   - Usar GET y observar cómo cambia la URL
   - Comparar las diferencias entre GET y POST

Ejercicio 3: Servidor de Archivos Estáticos

Objetivo: Servir archivos estáticos como CSS, JS e imágenes.

1. Crea una estructura de directorios:

   ```
   /static
   /css (estilos.css)
   /js (script.js)
   /img (logo.png)
   index.html
   ```

2. Implementa un servidor que:
   - Sirva los archivos estáticos usando `http.FileServer`
   - Maneje rutas incorrectas adecuadamente
   - Establezca los `Content-Type` correctos para cada tipo de archivo
3. Extiende el servidor para agregar compresión GZIP a las respuestas

Ejercicio 4: Análisis de Peticiones HTTP

Objetivo: Examinar peticiones y respuestas HTTP.

1. Usando `curl`, realiza peticiones a tu servidor y analiza:
   - Las cabeceras de solicitud y respuesta
   - Los códigos de estado HTTP
2. Ejercicios específicos:

1. Obtener solo las cabeceras: `curl -I http://localhost:8080/`
2. Enviar datos por POST: `curl -X POST -d "user=test&pass=123" http://localhost:8080/login`
3. Modificar el User-Agent: `curl -A "MiNavegador/1.0" http://localhost:8080/`
4. Crea un handler que muestre:
   - Método HTTP utilizado
   - Cabeceras recibidas
   - Parámetros GET/POST
   - Dirección IP del cliente

Recursos Adicionales

- Documentación net/http[1]
- MDN HTTP[2]
- Go by Example[3]
- HTTP Status Codes[4]

Trabajo de Cursada: Mi Primera Aplicación Web

Objetivo: Definir el dominio de la aplicación a desarrollar durante el curso y crear un servidor web inicial.

A lo largo de los trabajos prácticos, desarrollarás de manera incremental una aplicación web CRUD (Crear, Leer, Actualizar, Borrar). En este primer paso, sentarás las bases.

1. **Elección del Dominio**:
   - Elige un tema para tu aplicación. Debe ser algo simple que se pueda representar con pocas entidades principales. Ejemplos:
     - Una lista de tareas (To-Do List).
     - Un catálogo de tus películas favoritas.
     - Una lista de compras.
     - Un registro de libros leídos.
   - Define qué información guardarás para cada elemento (por ejemplo, para una tarea: `título`, `descripción`, `estado`).
2. **Página de Presentación**:
   - Crea un archivo `index.html` simple.
   - En este archivo, describe brevemente la aplicación que planeas construir. Incluye un título (`<h1>`) y un párrafo (`<p>`) con la descripción.
3. **Servidor Web Básico**:
   - Crea un servidor en Go (`main.go`) que sirva tu archivo `index.html` cuando se acceda a la ruta raíz (`/`).
   - Asegúrate de que el servidor escuche en el puerto `8080` y maneje correctamente el `Content-Type` del archivo HTML.

[1] https://pkg.go.dev/net/http
[2] https://developer.mozilla.org/es/docs/Web/HTTP
[3] https://gobyexample.com/
[4] https://httpstatuses.com/