REPORTE

Link de portafolio en GitHub: Asignatura Paradigmas

Link de portafolio en GitHub Pages (página estática): Asignatura Paradigmas

1. Markdown

Markdown es un lenguaje de marcado ligero creado originalmente por John Gruber y Aaron Swartz en 2004. Su propósito fundamental es maximizar la legibilidad y la publicabilidad, permitiendo que un documento escrito en texto plano tenga una estructura clara que pueda convertirse automáticamente a HTML (HyperText Markup Language).

Aplicaciones y Uso

El uso de Markdown se ha estandarizado en la industria tecnológica debido a su versatilidad:

- Documentación Técnica: Es el estándar para archivos README en plataformas como GitHub y GitLab.
- Generadores de Sitios Estáticos: Herramientas como Hugo, Jekyll y Quartz procesan archivos Markdown para generar sitios web rápidos.
- Gestión de Conocimiento: Aplicaciones como Obsidian, Notion y Logseq lo utilizan como base para el almacenamiento de notas.
- Comunicación: Plataformas como Slack, Discord y Microsoft Teams implementan subconjuntos de su sintaxis para formatear mensajes.

Sintaxis Fundamental

La sintaxis de Markdown utiliza caracteres de puntuación comunes para indicar el formato.

1. Encabezados Se utilizan almohadillas (#) al inicio de la línea. El número de almohadillas determina el nivel del encabezado (del 1 al 6).

2. Énfasis y Formateo

- Negrita: **texto** o **texto**
- Cursiva: *texto* o *texto*
- ~~Tachado:~~ ~~texto~~

3. Listas

- Desordenadas: Se utilizan asteriscos (*), guiones (-) o signos de suma (+).
- Ordenadas: Se utilizan números seguidos de un punto (1., 2.).

4. Vínculos e Imágenes La estructura para enlaces utiliza corchetes para el texto y paréntesis para la URL. Para las imágenes, se antepone un signo de exclamación.

- Enlace: Nombre
- Imagen: Texto alternativo

5. Bloques de Código Para fragmentos de código en línea se usan comillas simples invertidas (`). Para bloques de código multilínea, se utilizan tres comillas invertidas (```).

2. ¿Qué es Git y GitHub?

Aunque suelen mencionarse juntos, cumplen funciones distintas pero complementarias:

- Git: Es un sistema de control de versiones local. Como una "máquina del tiempo" para el código. Permite guardar capturas (commits) de proyecto, permitiéndo regresar a versiones anteriores si algo sale mal.

- GitHub: Es una plataforma de alojamiento en la nube que utiliza Git. Es donde se guarda los proyectos para que otros los vean, colaboren o simplemente para tener un respaldo seguro fuera de la computadora.

3. Comandos Esenciales de Git

Para dominar Git, primero se debe entender el flujo de trabajo básico. Estos son los comandos mas usados:

- `git init` Inicializa un nuevo repositorio de Git en tu carpeta local.
- `git clone [URL]` Descarga un proyecto existente desde la nube a tu PC.
- `git status` Te dice qué archivos has modificado y cuáles están listos para guardarse.
- `git add [archivo]` Prepara los archivos (los pone en el Staging Area) para el siguiente paso.
- `git commit -m "mensaje"` Guarda permanentemente los cambios con un comentario descriptivo.
- `git branch -M main` Cambia el nombre de la rama principal a "main" (estándar actual).
- `git remote add origin [URL]` Conecta tu repositorio local con tu repositorio en GitHub.
- `git push -u origin main` Sube tus cambios locales a la nube (GitHub).

4. Cómo crear un repositorio y subirlo a la nube

Paso A: En GitHub

1. Inicia sesión en tu cuenta de GitHub.
2. Hazcl ic en el botón "+" (arriba a la derecha) y selecciona New repository.
3. Ponle un nombre a tu proyecto y haz clic en Create repository.
4. Copia la URL que te aparece (ejemplo: https://github.com/tu-usuario/tu-proyecto.git).

Paso B: En computadora (Terminal o CMD)

1. Inicializar: git init
2. Preparar archivos: git add . (El punto significa "todos los archivos").
3. Primer guardado local: git commit -m "Mi primer commit"
4. Conectar con la nube: git remote add origin https://github.com/tu-usuario/tu-proyecto.git
5. Subir la información: git push -u origin main

5. ¿Qué es Hugo y GitHub Actions?

Para publicar una web hoy en día, no se necesitas un servidor complejo. Solo se ocupa de estas dos herramientas:

- Hugo: Es un Generador de Sitios Estáticos (SSG). Escribes contenido en lenguaje sencillo (Markdown), y Hugo lo transforma instantáneamente en una página web profesional (HTML/CSS) extremadamente rápida.
- GitHub Actions: Es un motor de automatización. Permite ejecutar "recetas" (workflows) cada vez que subes código. En este caso, le diremos: "Oye, cada vez que suba un cambio, usa Hugo para construir mi web y publícala en internet".

6. Crear un sitio estático con Hugo

1. Instalar Hugo: (Depende de tu sistema, ej. brew install hugo o descargando el binario).
2. Crear el sitio: `hugo new site mi-blog cd mi-blog`
3. Añadir un tema: `git submodule add https://github.com/theNewDynamic/gohugo-theme-ananke.git themes/ananke`
4. Crear primer post: `hugo new posts/mi-primer-post.md`
5. Verlo en vivo localmente: `hugo server -D` y entra a localhost:1313.

7. Subir la información a GitHub

1. Crea un archivo llamado `.gitignore` en la raíz y escribe dentro: `/public`.

8. Configurar GitHub Actions para publicar en GitHub Pages

Paso 1: Crear el archivo de Workflow En tu carpeta del proyecto, crea esta ruta de carpetas: `.github/workflows/` y dentro crea un archivo llamado `hugo.yaml`.

Paso 2: El contenido del Workflow Copia una plantilla oficial de Hugo para GitHub Actions. Básicamente, este archivo le dice a GitHub:

- Usa Ubuntu.
- Instala Hugo.
- Ejecuta el comando hugo.
- Toma el resultado y súbelo a GitHub Pages

Paso 3: Configuración en el Navegador (GitHub.com)

1. Ve a tu repositorio en GitHub.
2. Haz clic en Settings (Configuración) > Pages.

3. En la sección Build and deployment, cambia la fuente (Source) de "Deploy from a branch" a "GitHub Actions".