

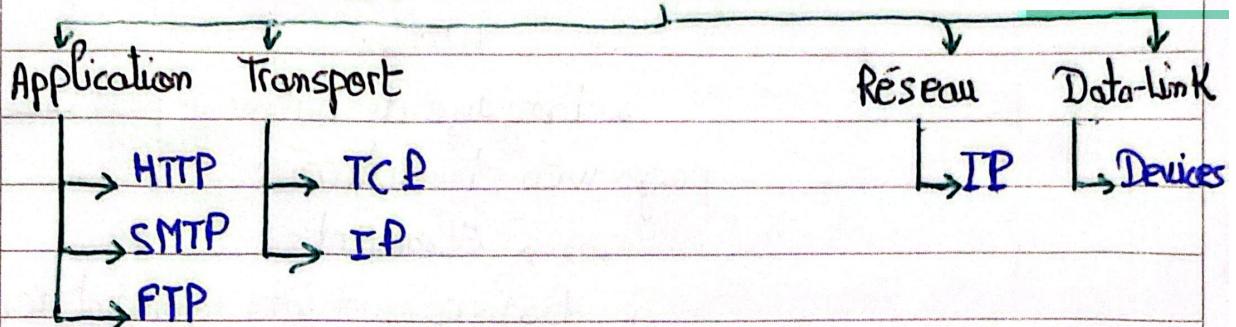
→ 101 Web dev :

→ 100 concepts you should know:

→ Internet:

réseau mondial de communication qui connecte des millions d'ordi et de dispositifs.

→ Internet Protocol Suite:



→ Web

* Réseau mondial d'information avec pages interconnectées, accessible sur Internet avec HTTP

→ HTTP:

* Protocole pour accéder aux pages web sur le réseau mondial.

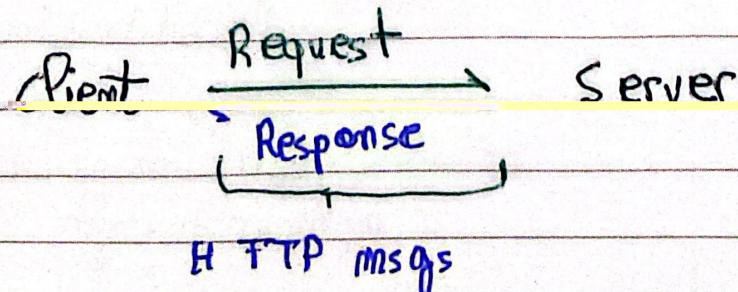
→ URL:

* Adresse unique pour accéder à une page spécifique en ligne.

→ Browser:

* Application pour explorer et afficher des pages web sur internet

→ Client, Server, Req, Res



→ Domaine name:

- * Nom unique → adresse sur internet

→ Registrar:

- * Entité qui enregistre et gère les noms de domaine internet

→ DNS:

- * Système qui traduit les noms de domaine en IP

→ HTML:

- * Langage de balisage pour créer des pages web structurées.

→ Elements:

- * Balises ou objets formant le contenu d'une page web HTML

→ ATTRibutes:

- * Pour ajouter des propriétés aux Elements

→ Anchor:

- * ` awesome web `

→ DOM :

- * La structure d'une page web.

→ Div :

- * Division utilisée pour grouper certains éléments HTML.

→ CSS

- * l'apparence des éléments HTML

→ Inline Style:

- * `<div style="color:red;"> </div>`

→ Selectors:

- * Expression qui cible les éléments pour leur appliquer des styles.

→ Class :

- * Attribut en HTML et CSS, utilisé pour regrouper des éléments similaires.

→ External STYLESHEET

- * style.css ; link:css

→ BOX Model :

- * Concept en CSS décrivant la structure d'une box entourant chaque élément.

→ Display :

BLOCK

INLINE

→ Responsive Layout:

- * conception de page ajustée selon les appareils

→ Media Query:

- * Appliquer des styles en fonction des caractéristiques de l'appareil.

→ flex Box:

- * méthode CSS pour organiser facilement des éléments dans une page web de manière flexible & responsive.

→ Grid Layout:

- * système CSS qui permet de créer des mises en pages complexes et réactives en définissant des grilles sur des pages web.

→ calc():

- * effectuer des calculs.

→ Custom Properties:

- * Global var:

:root {

color: white;

}

→ JavaScript :

- * L.P => créer des interactions dynamiques sur les pages web.

→ Script Tag :

* <script>

```
const hello = 'hi';
alert(hello);
```

</script>

→ Defer :

- * retarder l'exécution de js => la page soit complètement chargée.

→ Var

Let
↓
mutable

Const
↓
immutable

→ Js => Dynamically Typed..

- * Type de var est déterminé auto

→ TypeScript => Static Typed.

- * Let hello: string;

→ Events :

- * des interactions d'utilisateur, comme click ou taper, qui activent des réponses codées dans une app.

→ Browsers API :

- * Fourmissent des outils pour interagir avec le web et OS d'utilisateur. par exemple:

→ Geolocation API.

→ Local Storage.

→ Fetch API

→ Event Listener:

↳ un mécanisme qui surveille et réagit à des actions spécifiques

- ↳ des clics (btn.onclick ...)
- ↳ des pressions

→ Array

* const arr = [1, 2, 3];

→ Object

* const obj = {
 key: "value",
 foo: "bar"

}

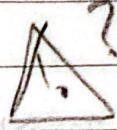
→ Primitives:

↳ des types de données les plus simples qui représentent une seule valeur, ils ne peuvent pas être changés une fois créés.

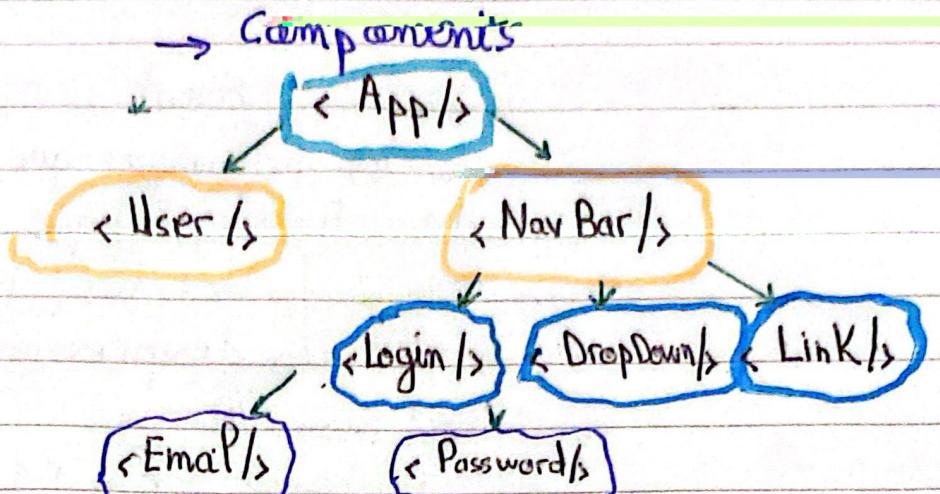
- String
- Number
- BigInt : Grand nombres
- Boolean
- Undefined
- Symbol
- Null

→ Prototypal inheritance:

* système où les objets héritent des propriétés et méthodes d'autres objets via une chaîne de prototypes



Dakhi Block crevina
Frameworks



→ Composants
 → Elements réutilisables qui permettent de structurer et de construire l'interface.

→ Declarative:

- * Consiste à décrire ce que vous voulez accomplir, plutôt que spécifier comment le faire étape par étape.

≠

→ Imperative:

→ Node.js:

- * Env d'exécution de js côté Server.

→ V8 Engine:

- * le moteur js opensource de Google utilisé dans Node.js et Chrome.

→ Event Loop:

- * mécanisme qui permet à Node.js de gérer de manière asynchrone I/O

→ Async:

- * Exécution des tâches indépendantes de manière non-bloquante, ce qui permet à un programme de continuer à fonctionner pendant que les autres tâches sont en cours.

→ SSR (Frameworks):

- * génère la page web sur le serveur avant de l'envoyer au navigateur

→ HTTP Method:

- * sont des actions utilisées pour interagir avec les ressources web

→ Get: récupérer les données.

→ Post: soumettre des données.

→ Put: mettre à jour ou créer une ressource si elle n'existe pas.

→ Patch: appliquer les modifications partielles

→ Delete

→ Head

→ Option

→ Trace

→ Status Code:

→ SPA:

- * app qui fonctionne en chargeant une seule page HTML et en mettant à jour dynamiquement son contenu sans recharger la page entière.

→ JSON:

- * un format de données lisible et utilisable pour représenter et éditer des données structurées.

→ SSG

- * consiste à créer un site web en préparant à l'avance des fichiers HTML, CSS, JS statiques, ce qui améliore la rapidité et la sécurité.

→ Hydratation:

* permet de rendre interactives les pages statiques en réactivant les fonctionnalités JS une fois qu'elles sont chargées dans browser.

→ FCP & TTI:

First Contentful Paint → FCP : mesure le moment où le premier élément visible apparaît à l'écran.

Time To Interact → TTI : mesure le temps nécessaire que la page devienne totalement interactive.

→ Module bundler:

→ outils de dev ⇒ regrouper et organiser les modules JS en un seul fichier.

→ Linter:

→ " " " ⇒ analyser le code d'un programme et identifier les erreurs.

Exemple ⇒ .eslintrc.js

→ User auth:

→ Processus de vérification de l'id d'un user → permettre d'accéder à un sys.

→ Web Server:

→ Apache, NGINX

→ Cloud:

→ Containers:

→ Env fine tuning l'app & dépendances
batch shell 3line deployment & batch than Kima
Haban 3rdma f Local.

→ Docker

→ Web 3: