Séquence 1 – Introduction aux IHM — Durée : 1h15

TP – Duels d'interfaces

Nom : ..... **Groupe :** .....

Objectifs de la séance

- Mesurer par la pratique les forces et faiblesses de la **ligne de commande** (CLI) face à l'**interface graphique** (GUI).
- Traduire en CSS deux courants visuels opposés – **flat design** et **skeuomorphisme** – et en analyser l'effet sur l'affordance et l'accessibilité.

Déroulé indicatif : Partie A (30 min) – Partie B (40 min) – bilan collectif (5 min).
Rendu : un dossier `tp1_nom1_nom2` contenant `conclusion.txt` et `formulaires.html`, déposé sur un dépot `git` de nom `IHM-BUT2-2026` en fin de séance.

Partie A – CLI vs GUI : le duel du renommage *30 min*

Mise en place (3 min). Créez un dossier `tp_cli_vs_gui` contenant 15 fichiers vides : 10 fichiers `.txt` et 5 fichiers `.jpg`. Pour gagner du temps, utilisez les commandes ci-dessous (vous les comprendrez à la fin du TP).

Linux / macOS (bash, zsh) :

```
mkdir tp_cli_vs_gui && cd tp_cli_vs_gui
touch rapport_v{1..5}.txt notes_{a..e}.txt
touch photo-0{1..5}.jpg
ls
```

La tâche : renommer **tous** les fichiers `.txt` en `.md` et **tous** les fichiers `.jpg` en `.png`.

Q1. Manche 1 – GUI. Chronométrez-vous en réalisant la tâche **uniquement** avec l'explorateur de fichiers de votre système (clic droit → Renommer, F2…). Notez votre temps et **chaque difficulté** rencontrée.

.....

.....

Q2. Avant même de commencer, avez-vous dû modifier un réglage de l'explorateur ? Lequel, et pourquoi est-il désactivé par défaut ? Reliez votre réponse à un principe du cours.

.....

.....

Q3. Manche 2 – CLI. Remettez les fichiers dans leur état initial (supprimez le dossier et relancez la mise en place). Ouvrez un terminal dans le dossier et trouvez – documentation, `man`, recherche web autorisés – la ou les commandes qui réalisent la tâche **en une seule fois**. Chronométrez **recherche incluse**.

.....

.....

.....

Q4. Relancez la mise en place une dernière fois et refaites la manche 2 maintenant que vous connaissez la commande. Nouveau temps ?

. . . . . .

Q5. Estimez le temps nécessaire pour chaque méthode s'il y avait **1 500 fichiers**. À partir de combien de fichiers la CLI devient-elle rentable *la première fois* ? Et les fois suivantes ?

. . . . . .

. . . . . .

Q6. Ouvrez une des images renommées : `photo-01.png` est-elle maintenant une image PNG ? Vérifiez avec la commande `file` (Linux/macOS). Qu'en déduisez-vous sur ce que « montre » une interface ?

. . . . . .

. . . . . .

Q7. Une erreur de frappe dans votre commande aurait pu renommer **tous** les fichiers en un seul nom, en écrasant les autres. Comment la GUI vous protège-t-elle de ce risque ? Comment se protéger en CLI ?

. . . . . .

. . . . . .

Coup de pouce pour la manche 2

En bash, une boucle `for f in *.txt; do ...; done` parcourt les fichiers ; l'expression `${f%.txt}` donne le nom de `f` **sans** son suffixe `.txt`. En PowerShell, cherchez du côté de `Get-ChildItem` et `Rename-Item`.

☞ Livrable

Un fichier `conclusion.txt` structuré ainsi :

1. les commandes utilisées (copiées depuis votre terminal) ;
2. vos temps : GUI, CLI première fois (recherche incluse), CLI seconde fois ;
3. **analyse (10 lignes max)** : dans quel scénario la GUI est-elle supérieure ? Dans quel scénario la CLI est-elle imbattable ? Utilisez au moins deux notions du cours (découvrabilité, rappel/reconnaissance, feedback, scriptabilité, ...).

Partie B – Voyage dans le temps visuel : Flat vs Skeuomorphisme *40 min*

Récupérez sur mon git le fichier de départ `formulaires.html` (reproduit ci-dessous). Il contient **deux fois le même formulaire de connexion** ; seule la classe de la `<section>` change. Vous ne devez modifier **que le CSS** (dans la balise `<style>`).

```
<!DOCTYPE html>
<html lang="fr">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Flat vs Skeuo</title>
  <style>
    body    { font-family: system-ui, sans-serif; margin: 0; padding: 2rem; }
    section { max-width: 360px; margin: 0 auto 3rem; padding: 2rem; }
    form    { display: flex; flex-direction: column; gap: .5rem; }
    /* ===== Version 1 : Flat design ===== */
```

```
    /* ===== Version 2 : Skeuomorphisme (look 2010) ===== */

  </style>
</head>
<body>
  <section class="flat">
    <h2>Version 1 - Flat</h2>
    <form>
      <label for="email1">Email</label>
      <input type="email" id="email1" placeholder="prenom.nom@exemple.fr">
      <label for="mdp1">Mot de passe</label>
      <input type="password" id="mdp1">
      <button type="submit">Se connecter</button>
    </form>
  </section>
  <section class="skeuo">
    <h2>Version 2 - Skeuomorphisme</h2>
    <form> <!-- identique, avec les id email2 et mdp2 --> </form>
  </section>
</body>
</html>
```

Q8. Version 1 – Flat (15 min). Stylisez la section `.flat` en respectant **scrupuleusement** les codes du flat design :

- **aucune** ombre (`box-shadow`, `text-shadow`), aucun dégradé, aucune texture ;
- couleurs **unies**, une seule couleur d'accent pour le bouton ;
- typographie claire, hiérarchie par la **taille** et la **graisse** uniquement ;
- un état `:hover` et un état `:focus-visible` visibles sur le bouton et les champs.

Q9. Version 2 – Skeuomorphisme (20 min). « Dé-makez » la section `.skeuo` façon 2010 :

- un bouton en **relief 3D** : dégradé vertical, reflet clair en haut (`inset`), ombre portée ;
- un état `:active` où le bouton semble **enfoncé** ;
- des champs **creusés** dans la matière (ombre intérieure `inset`) ;
- une **texture** de fond (lin, bois, papier…) **générée en CSS** – pas d'image externe.

Q10. Analyse (5 min). Répondez dans un commentaire HTML en bas du fichier :

(a) Montrez les deux versions à votre voisin · e sans rien dire : quel bouton « a l'air » le plus cliquable ? Quel principe du cours cela illustre-t-il ?

(b) Avec l'outil d'inspection du navigateur (onglet *Accessibilité* ou survol de la couleur), relevez le **ratio de contraste** du texte du bouton dans chaque version. Le minimum WCAG pour du texte normal est **4,5 :1**. Résultat ?

(c) Pourquoi le neumorphisme (vu en cours) a-t-il été abandonné, alors qu'il utilise les mêmes outils CSS que votre version 2 ?

Boîte à outils CSS

```
/* Dégradé vertical avec cassure "verre" au milieu */
background: linear-gradient(#7fb2f0 0%, #3b7bd1 50%, #2a63b5 51%, #2f6fc4 100%);
/* Plusieurs ombres séparées par des virgules ; "inset" = ombre intérieure */
box-shadow: inset 0 1px 0 rgba(255,255,255,.6), 0 2px 3px rgba(0,0,0,.4);
/* Texture "lin" : deux rayures fines croisées superposées à une couleur */
background-color: #d8cbb0;
background-image:
  repeating-linear-gradient(45deg,  rgba(255,255,255,.08) 0 2px, transparent 2px 4px),
  repeating-linear-gradient(-45deg, rgba(0,0,0,.06)       0 2px, transparent 2px 4px);
/* Texte "gravé" */
```

```
text-shadow: 0 1px 0 rgba(255,255,255,.6);
```

Bilan collectif (5 min)

Deux ou trois binômes présentent leur temps (Partie A) et leur version skeuomorphique (Partie B). Question de synthèse : *« Une interface plus "jolie" est-elle plus utilisable ? »*

Pour aller plus loin (travail personnel facultatif)

— **Retour vers les 90s.** Créez une page personnelle sur le thème de votre choix avec les contraintes de 1998 : mise en page **uniquement** en `<table>` (ni Flexbox ni Grid), au moins un GIF animé, polices « web safe » (Comic Sans MS, Times New Roman, Arial), un faux compteur de visites, un fond en mosaïque (`background-repeat`). Puis listez ce qui serait inaccessible avec un lecteur d'écran.

— **Réimagination mobile-first.** Choisissez Excel, Photoshop ou VS Code et dessinez l'écran d'accueil et l'écran d'édition d'une première version mobile. Quelles fonctions sacrifiez-vous ? Comment remplacer le clic droit et la sélection précise ? *(Excellent échauffement pour la séquence 2.)*

— **Gardien des guidelines.** Dans une appli de votre téléphone, trouvez un respect parfait, un écart volontaire et un point améliorable vis-à-vis des *Human Interface Guidelines* (iOS) ou de *Material Design* (Android). Captures annotées.

— **Prototypage conversationnel.** Écrivez sous forme de diagramme de flux le dialogue entre un utilisateur et un assistant pour réserver un billet de train, y compris les ambiguïtés (« ce soir ») et les corrections (« non, Lyon, pas Lille »).

— **Le développeur augmenté.** Faites générer par un assistant IA une carte produit HTML/CSS, puis rédigez un `CRITIQUE.md` : sémantique, accessibilité, ce que vous avez dû corriger, temps gagné.