Les langages d'une page web

Sylvain Tenier, Karim Hammoudi, Vincent Derrien

Département TIC - Esigelec

2015 - semestre 7

Plan

1. HTML : structure et hyperliens
2. Présentation avec CSS
3. Manipulation avec Javascript
4. Les bonnes pratiques à appliquer

Technologies composant une page web

- Une page web est composée à partir d'un ou plusieurs fichiers contenant 3 langages

HTML (HyperText Markup Language)

Définit la *structure* de la page sous la forme de balises autour du contenu et les *liens* vers d'autres resources (navigation et intégration)

CSS (Cascading StyleSheets)

Définit la *présentation* de la page (positionnement, couleurs, polices, ...)

Javascript

Permet de *manipuler* la page une fois chargée (animations, gestion d'événements, modifications, ...)

Première page HTML

Page d'accueil de l'Esigelec simplifiée

```
1  <!DOCTYPE html>
2  <html>
3    <head>
4      <meta charset="utf-8">
5      <title>L'Esigelec</title>
6    </head>
7    <body>
8      <h1>L'école</h1>
9      <img src="logo_esigelec.png" alt="
       logo">
10     <p>L'ESIGELEC c'est une formation é
       quilibrée entre acquisition des
       technologies, formation humaine et
        développement du potentiel [...]<
       /p>
11   </body>
12 </html>
```

HyperText *Markup* Language

Markup $\Longleftrightarrow$ balisage

1. Un document HTML est un fichier *texte* auquel sont ajoutées des *balises* qui définissent sa structure et sa sémantique
2. Ce fichier est *composé* avec un *éditeur de texte* (pas un traitement de texte !) tel que notepad++, gedit, textmate, vim, emacs, Sublime Text
3. Le résultat est *publié* sur un *serveur web* fourni par des outils tels que EasyPHP ou MAMP (en développement)
4. Il est *interprété* par le navigateur web dont le rôle est de *charger* la page depuis le serveur puis de l'*afficher*
   - Un navigateur web est également capable *d'ouvrir* un fichier HTML local sans passer par un serveur

   Cette pratique est à proscrire dans le cadre du module

Syntaxe HTML

Règles de balisage d'un document

- Une balise ouvrante est créée à partir du nom de l'élément HTML entouré par des chevrons `< >`
  - Ex : `<head>`, `<body>`, `<p>`, `<h1>`, `<img>`
- L'ouverture d'une balise déclare une zone où sa sémantique s'applique
  - Ex : `<p>` déclare le début d'un paragraphe
- La fin de la zone est marquée par une balise fermante
  - Ex : `</head>`, `</body>`, `</p>`, `</h1>`
- Certaines balises sont dites "autofermantes". On peut alors ignorer la balise de fermeture
  - Ex : `<img>`, `<br>`, `<hr>`, `<meta>`

Attributs HTML

- Une balise ouvrante peut contenir des informations additionnelles, appelées *attributs*
- Les attributs sont composés de leur *nom* et de leur *valeur*
- Exemple : `<img src="logo_esigelec.png" alt="logo">`
  - L'attribut `src` de la balise `img` indique l'adresse de l'image à intégrer à la page
  - L'attribut `alt` permet l'affichage d'un texte alternatif en cas d'erreur de chargement ou pour des raisons d'accessibilité (lecteurs pour malvoyants)

Page HTML valide minimale

doctype, en-tête et corps

```
1 <!DOCTYPE html>
2 <html>
3  <head>
4   <meta charset="utf-8">
5   <title></title>
6  </head>
7  <body>
8  </body>
9 </html>
```

- Le doctype indique la version d'HTML utilisée
- `<head>` définit la zone des métadonnées
- `<body>` englobe tout le contenu affiché

Application 1 : servir sa première page

Edition

1. Récupérez sur ENT l'exemple minimal et ouvrez le dans `Notepad++`
2. Ajoutez un titre entre `<title>` et `</title>` et un contenu entre `<body>` et `</body>`
3. Enregistrez le fichier dans dossier sur votre lecteur `X:` et renommez le "page1.html"

Publication

1. Lancez `EasyPHP` et vérifiez que l'icône est apparue dans la zone de notifications
2. Copiez-collez le dossier contenant votre fichier "page1.html" dans le dossier `C:\www\`
3. Saisissez `127.0.0.1` dans la barre d'adresse de votre navigateur

Bilan de l'application 1

Méthodologie

1. Vous *éditez* vos pages web dans un éditeur de texte
2. Vous *enregistrez* dans un dossier de travail que vous conservez entre les séances (par exemple, dans le lecteur réseau `X:`)
3. Vous *publiez* votre travail dans le dossier racine du serveur web (`C:\www\` pour EasyPHP sur les machines de l'École)
4. Vous *visualisez* avec votre navigateur à l'adresse `http://127.0.0.1`

HyperText Markup Language

HyperText $\Longleftrightarrow$ liens entre ressources

- Les ressources sont reliées entre elles
  1. Par des liens vers d'autres pages ou ressources
     - `<a href="autre_page.html">voir page</a>`
     - `<a href="doc.pdf">télécharger</a>`
  2. Par l'intégration de ressources (image, vidéo, son, flash...) supplémentaires à l'intérieur d'une page
     - `<img src="logo_esigelec.png" alt="logo">`
     - `<object type="application/x-shockwave-flash" data="anim.swf" width="300" height="200"> <param name="movie" value="anim.swf" /> <param name="quality" value="high" /> </object>`

Liens absolus, relatifs et ancres

1. Un lien *absolu* est défini par un URI complet
   - Il doit être utilisé vers les ressources *externes au serveur*
   - ex : `<a href="http://www.google.fr">`
2. Un lien *relatif* est défini par un chemin au format unix
   - La ressource doit être située sur le même serveur
   - Exemples :
     - `<a href="tests/page2.html">`
     - `<img src="../img/logo_esigelec.png">`

Ne jamais utiliser de lien absolu vers vos propres ressources

3. Un lien *ancre* est défini par un # suivi d'un identifiant
   - Il fait référence à un endroit précis d'une page : l'élément HTML possédant un attribut `id` dont la valeur est l'identifiant
   - ex : `<a href="#par3">` pointe vers `<p id="par3">`
   - Un lien relatif ou absolu peut être complété par une ancre

Application 2 : création de liens

Édition

1. À partir du modèle minimal, créez un nouveau fichier “page2.html” dans le même dossier que la première page
2. Ouvrez “page2.html” dans Notepad++ et insérez le code

```
<a href="http://www.esigelec.fr">Esigelec</a>
```

3. Dans le corps de votre première page “page1.html”, ajoutez le code `<a href="page2.html">la page 2</a>`

Publication

1. Republiez votre code dans le répertoire racine d’EasyPHP
2. Rafraichissez “page1.html” dans votre navigateur et cliquez sur le lien

HTML : les points clés

- Un document HTML est un document texte auquel sont intégrées des balises qui :
  1. définissent la structure du document
  2. permettent d'intégrer d'autres ressources
  3. permettent de relier le document à d'autres ressources
- La création d'un document HTML se déroule en 2 étapes
  1. La création avec un éditeur de texte
  2. La publication dans un serveur web
- Le navigateur web accède au document déployé sur le serveur web, l'interprète et l'affiche à l'utilisateur

Plan

1 HTML : structure et hyperliens

2 Présentation avec CSS

3 Manipulation avec Javascript

4 Les bonnes pratiques à appliquer

Gestion de la présentation

- À chaque élément HTML, le navigateur applique un style par défaut
- Le langage CSS permet de personnaliser l'apparence de sa page
- Sa syntaxe est différente du HTML !
- Le CSS peut être :
  1. importé depuis des fichiers .css (meilleure pratique)
  2. intégré à la page entre des balises `<style>` et `</style>` (idéal pour les tests)
  3. défini comme attribut dans un élément HTML (à proscrire)

CSS intégré dans le HTML

```
1  <!DOCTYPE html>
2  <html>
3    <head>
4      <meta charset="utf-8">
5      <title>Ma page avec du CSS intégré</title>
6      <style>
7        body {
8        background-color: #d2b48c;
9        margin-left: 20%;
10       margin-right: 20%;
11       border: 2px dotted black;
12       padding: 10px 10px 10px 10px;
13       font-family: sans-serif;
14       }
15     </style>
16   </head>
17   ...
18  </html>
```

(aparté historique)

- Depuis HTML 4, les CSS sont l'unique moyen de fournir la présentation
- Jusqu'en juillet 1997 (HTML 3.2) la présentation s'effectuait avec des balises HTML
  - `<b>` pour gras, `<i>` pour italiques, `<font>` pour la police...
- Certains éléments HTML sont réutilisés avec une sémantique différente
  - `<b>` et `<i>` indiquent désormais que le contenu est important, pas comment il doit être affiché
- D'autres sont invalides (ex : `<font>`)

Pensez à valider votre code systématiquement[1]

[1] https://jigsaw.w3.org/css-validator

Application 3 : styles CSS en pratique

Edition

1. Créez un fichier “style.css” dans votre dossier
2. Copiez-coller le code situé entre les balises `<style>` et `</style>` de l’exemple (sans inclure les balises)
3. Ajoutez `<link rel="stylesheet" href="style.css">` dans la partie `<head>` de vos fichiers “page1.html” et “page2.html”

Publication

1. Republiez l’ensemble de vos fichiers dans le répertoire racine d’EasyPHP
2. Rechargez votre page “page1.html” dans le navigateur web
3. Videz le cache du navigateur si nécessaire (`Ctrl + F5`)

Syntaxe et terminologie CSS

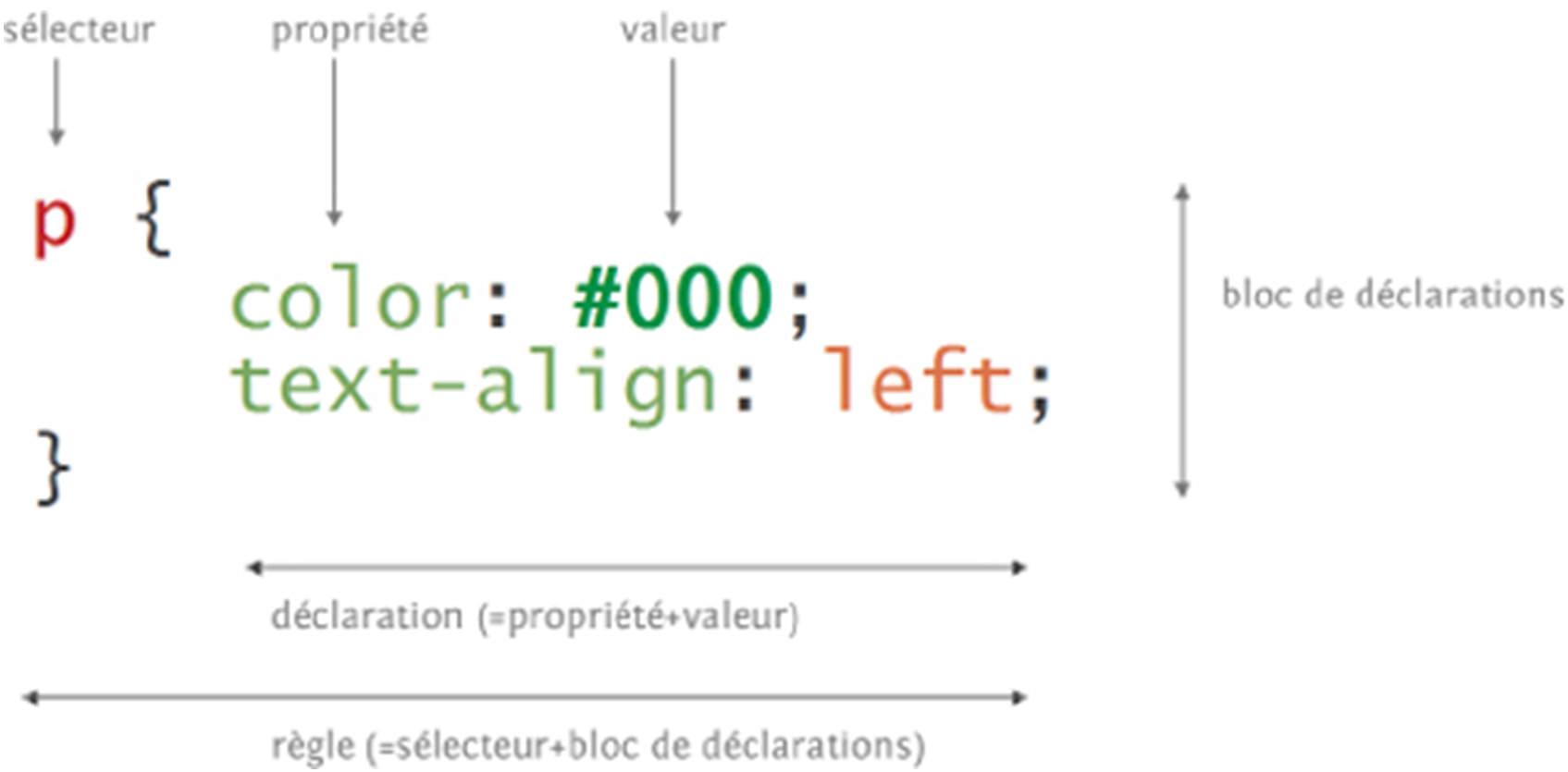
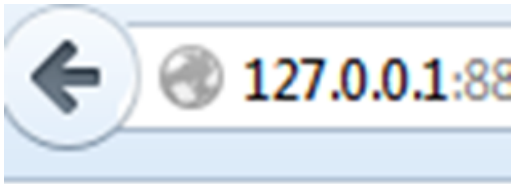


FIGURE : exemple de règle applicable à tous les paragraphes

Sélecteurs CSS (1) : sélecteur d'élément

```
1  <!DOCTYPE html>
2  <html>
3    <head>
4      <meta charset="utf-8">
5      <title>Sélection par élément</title>
6      <style>
7        p {color:red;}
8      </style>
9    </head>
10   <body>
11     <p> paragraphe 1</p>
12     <p> paragraphe 2</p>
13     <div>paragraphe 3</div>
14     <div>paragraphe 4</div>
15   </body>
16 </html>
```



paragraphe 1

paragraphe 2

paragraphe 3

paragraphe 4

Sélecteurs CSS (2) : sélecteur d'identifiant

```
1  <!DOCTYPE html>
2  <html>
3    <head>
4      <meta charset="utf-8">
5      <title>Sélection par identifiant</title>
6      <style>
7        #p1, #p3 {color: red;}
8      </style>
9    </head>
10   <body>
11     <p id="p1" >paragraphe 1</p>
12     <p id="p2">paragraphe 2</p>
13     <div id="p3" >paragraphe 3</div>
14     <div id="p4">paragraphe 4</div>
15   </body>
16 </html>
```

paragraphe 1

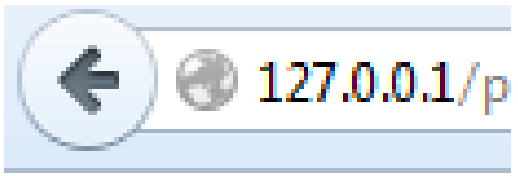
paragraphe 2

paragraphe 3

paragraphe 4

Sélecteurs CSS (3) : sélecteur de classe

```
1  <!DOCTYPE html>
2  <html>
3    <head>
4      <meta charset="utf-8">
5      <title>Sélection par classe</title>
6      <style>
7        .cb {color: blue;}
8      </style>
9    </head>
10   <body>
11     <p>paragraphe 1</p>
12   <p class="cb">paragraphe 2</p>
13     <div>paragraphe 3</div>
14     <div class="cb">paragraphe 4</div>
15   </body>
16 </html>
```



Sélecteurs CSS : résumé

Sélecteur d'élément `x`

sélectionne tous les éléments `x` de la page

Sélecteur d'identifiant `#ix`

sélectionne l'élément *unique* possedant l'attribut `id` correspondant

- Page non valide si attribut id dupliqué
- Code HTML : `<p id="ix">`

Sélecteur de classe `.cx`

sélectionne les éléments possédant l'attribut `class` correspondant

- Code HTML : `<p class="cx">`

Déclarations CSS : exemple 1

Gestion des couleurs

Propriétés

1. Couleur du texte : `color`
2. Couleur du fond : `background-color`

Valeurs possibles

1. 17 couleurs nommées (`white`, `red`, ...)
2. Code hexadécimal
   - Ex : blanc `#ffffff`, rouge `#ff0000`
3. Valeurs rgb en notation décimale [0-255]
   - Ex : blanc `rgb(255,255,255)`, rouge `rgb(255,0,0)`

- Référence :`https://developer.mozilla.org/fr/docs/CSS/Premiers_pas/Couleurs`

Déclarations CSS : exemple 2

Taille du texte

Propriété

- `font-size`

Valeurs possibles

1. absolue

   `xx-small | x-small | small | medium | large | x-large | xx-large`

2. relative `smaller | larger`
3. en pixels (ex : `16px`)
4. dynamique (ex : `10em`)

- Référence : `https://developer.mozilla.org/fr/docs/CSS/font-size`

Déclarations CSS : exemple 3

Autres propriétés applicables au texte

1. Police système : `font-family`
   - `https://developer.mozilla.org/fr/docs/CSS/font-family`
2. Italique : `font-style`
   - Valeurs : `normal | italic | oblique`
3. Gras : `font-weight`
   - Valeurs : `normal | bold | bolder | lighter | 100 | 200 | 300 | 400 | 500 | 600 | 700 | 800 | 900`
4. Hauteur de ligne : `line-height`
   - Valeurs :

     `normal | <nombre> | <longueur> | <pourcentage> | inherit`
5. Décoration : `text-decoration`
   - Valeurs : `none | [ underline || overline || line-through || blink ] | inherit`

Plan

1 HTML : structure et hyperliens

2 Présentation avec CSS

3 Manipulation avec Javascript

4 Les bonnes pratiques à appliquer

Qu'est-ce que Javascript ?

- "Le langage du navigateur"
  - Les scripts sont exécutés sur le terminal client
- Javascript permet notamment
  1. La création d'animations ergonomiques ou esthétiques
  2. La manipulation d'une page web *une fois chargée*
     - Généralement en réponse à des événements déclenchés par l'utilisateur
  3. Depuis "HTML5", la création d'applications complètes client/serveur
     - Exemple : e-services, logiciels de paie, ...

Atouts et faiblesses pour un étudiant Esigelec

Atouts

- Syntaxe proche de C/JAVA
- Librairies permettant la gestion facile des événements
- Multiples tutoriels disponibles

Faiblesses

- Langage interprété, absence de compilateur
- Typage faible
- Support différent selon les navigateurs
- De nombreux pièges pour une utilisation avancée
- Javascript peut être désactivé par l'utilisateur

Code en ligne ou externe

1. Intégration de code à la page HTML

```
1 <script>
2   function inlineScript() {
3     var message = "Hello, World!";
4     alert(message);
5   }
6   inlineScript();
7 </script>
```

2. Lien vers un fichier `.js` externe
   - Méthode recommandée pour respecter le principe de séparation des rôles

```
1 <script src="fichier.js"></script>
```

jQuery : une librairie pour unifier la manipulation

Tâches facilitées par jQuery (1)

1. Accéder à des éléments du document en réutilisant les sélecteurs CSS
   - `$('div')` sélectionne tous les éléments `div`
   - `$('#monid')` sélectionne l'élément possédant un attribut `id="monid"`
2. Modifier l'apparence d'un élément en ajoutant/supprimant des classes CSS
   - `$('div').addClass('x')` ajoute l'attribut `class="x"` à tous les éléments `div`
3. Modifier le *contenu* du document
   - `$('div#x').append('<p>nouveau contenu</p>')` ajoute un paragraphe à l'élément `<div id="x">`

jQuery : une librairie pour unifier la manipulation

Tâches facilitées par jQuery (2)

4. Réagir aux interactions de l'utilisateur
   - `$('button#b1').click(maFonction)` exécute la fonction `maFonction` lorsque l'utilisateur clique que le bouton `<button id="b1">`
5. Animer les transitions lors d'un changement
   - `$('#cache').slideUp()` fait apparaitre progressivement l'élément caché possédant l'attribut `id="cache"`
6. Charger du contenu supplémentaire sans rafraichir la page
   - `$('#contenant').load('contenu.html');` ajoute le contenu du fichier "contenu.html" à l'élément possédant l'attribut `id="contenant"`

Application 4 : Manipulation d'une page web
Partie 1/2

Dans un fichier .html

1. Récupérez sur ENT le fichier "applicationjs.html"
2. Ajoutez le code suivant entre `<head>` et `</head>`

```
1 <script src="https://code.jquery.com/jquery-2.1.4.min.js "></script
      >
2 <script src="monscript.js"></script>
```

Application 4 : Manipulation d'une page web
Partie 2/2

Dans un fichier "monscript.js"

1. Copiez-collez le code suivant

```
1  function quandCestPret(){
2    $('p').animate({
3    padding:'20px',
4    fontSize:'30px'
5  }, 2000);
6  $('div').toggleClass('cr');
7  }
8
9  $(document).ready(quandCestPret);
```

2. Publiez le code dans EasyPHP et ouvrez votre page HTML

Plan

1 HTML : structure et hyperliens

2 Présentation avec CSS

3 Manipulation avec Javascript

4 Les bonnes pratiques à appliquer

Trouver et valider l'information

2 étapes indissociables

1. Bien exploiter son moteur de recherche
   - 99,9% des problèmes ont déjà été traités !
   - Bien choisir ses mots clés
   - Exemple : "listes à puces html"
2. Vérifier la validité du résultat en utilisant une référence
   - CSS : `https://developer.mozilla.org/fr/docs/Web/CSS/Reference`
   - HTML : `https://developer.mozilla.org/fr/docs/Web/HTML/Element`

Validateurs automatiques

- Le W3C fournit des outils libres d'accès pour valider son travail
  1. HTML : `http://validator.w3.org/`
  2. CSS : `http://jigsaw.w3.org/css-validator/`
- Les navigateurs web modernes fournissent des outils de développement permettant également de valider son code Javascript.
- Utilisez ces outils à la manière d'un *Compilateur* qui vérifie la syntaxe de votre code

Commenter son code

- Un code doit toujours être commenté (cf. modules JAVA, projet,. . .)
- Les syntaxes sont dépendantes du langage

HTML `<!-- Un commentaire -->`

CSS `/* Un commentaire */`

Javascript `// Sur une ligne` ou `/* Plusieurs lignes*/`

Application 5 : validation

- Saisissez l'adresse du validateur (ou recherchez "validateur w3c")
- Cliquez sur "validate by direct input"
- Copiez-collez le contenu du fichier HTML de l'Application 2 "création de liens"
- Votre page est valide s'il n'y a pas d'erreurs (2 warnings sont normaux)

Liens utiles

- Les bases du langage HTML : `https://developer.mozilla.org/fr/docs/Web/Guide/HTML/Introduction`
- Premiers pas en CSS : `https://developer.mozilla.org/fr/docs/CSS/Premiers_pas`
- Pour aller plus loin en CSS : `http://www.alsacreations.com/apprendre/`
- Bases de jQuery : `https://learn.jquery.com/about-jquery/how-jquery-works/`