HTML 5

LI385 - Nouvelles Technologies de Web

HTML5

- Evolution (compatible) de HTML4
  - Les balises HTML4 restent valables en HTML5
    - Quelques notions obsolètes
- Nouvelles fonctionnalités :
  - Du langage HTML (balises, formulaires)
  - De Javascript
  - De CSS
- Compatibilité :
  - http://caniuse.com/

Plan

- Nouvelles balises sémantique/multimédia
- Formulaires
- Nouvelles APIs Javascript

CODE LIGHT ET NOUVELLES BALISES

Simplification du code

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="fr" >
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
  <link rel="stylesheet" type="text/css" href="design.css" />
  <script type=”text/javascript” src=script.js"></script>
</head>
```

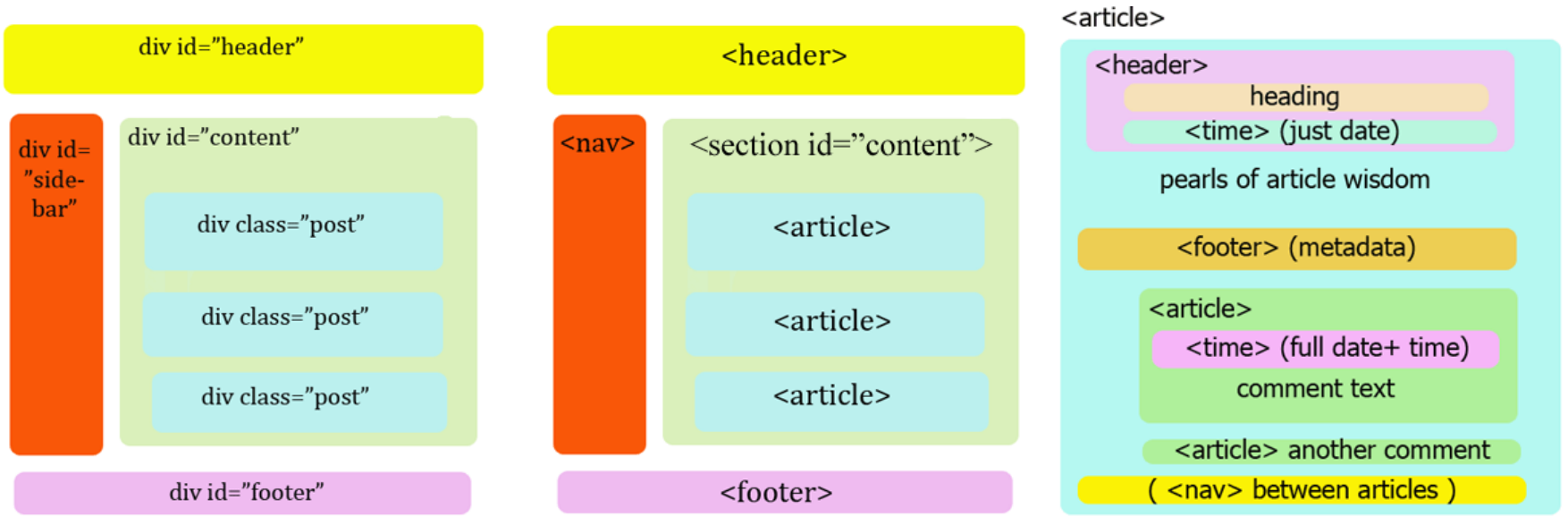
```
<!DOCTYPE html>
<html lang="fr">
<head>
  <meta charset="utf-8" />
  <link rel="stylesheet" href="design.css" />
  <script src="script.js"></script>
</head>
```

Nouvelles balises

- Pourquoi ne suffit pas ?

- Balises sémantiques :
  - <header> : en-tête
  - <footer> : pied-de-page
  - <nav> : navigation, type menu
  - <aside> : zone secondaire non liée au contenu principal de la page
  - <article> : portion de la page qui garde un sens même séparée de l'ensemble de la page (comme un article de blog par exemple)
  - <figure> : une illustration, par exemple image+légende+…

Structure globale



Barres de progrès

- Meter :
  - Représentation d'une valeur dans un intervalle
  - La couleur dépend de la valeur
  - <meter min="0" max="100" low="40" high="90" optimum="100" value="41">pas mal</meter>

- Progress :
  - Représente quelque chose en train d'augmenter
  - <progress value="75" max="100">3/4</progress>

Balises multimédia

- Audio :
  - Lecture de fichiers audio
  - MP3, Ogg, Wav
- Video :
  - Lecture de fichiers vidéo
  - MP4, Ogg, WebM
- Canvas :
  - Dessin
- Éventuellement prévoir des fallbacks (flash)

Balises audio/video

- Attributs audio :
  - autoplay
  - controls
  - loop
  - muted
  - preload
  - src
- Attributs video supplémentaires :
  - height, width
  - poster

- Événements associables :
  - onpause, onvolumechange, onseeking, …
  - http://www.w3schools.com/tags/ref_eventattributes.asp

FORMULAIRES

Nouveaux champs

- search : champ de recherche
- email : un ou plusieurs email
- url
- tel : pas de validation particulière
- datetime :
- date : champ de choix de date
- month
- week
- time
- datetime-local
- number
- range : choix d'un nombre
- color : color picker
- keygen : génération de clé publique

Nouveaux attributs

- Autocomplete : formulaire ou champ
  - Propose l'autocompletion
  - <input type="text" id="name" autocomplete="on" />
- Novalidate : formulaire
  - La validité du champ n'est pas vérifiée.
  - <form action="" novalidate>
- Autofocus : champ
  - Donne le focus par défaut à l'unique champ de ce type
  - <input type="email" id="email2" autofocus>
- Required : champ
  - Champ obligatoire
  - <input type="email" id="email" required>

Nouveaux attributs

- Placeholder : input
  - Champ prérempli mais avec contenu qui disparait
  - <input type="email" placeholder=mail@domain.com
- Readonly : input
- Multiple : input
  - Renseigner plusieurs valeurs (pas que pour un select)

Datalist

- Champ texte avec valeurs déroulantes prédéfinies
  - Possibilité de taper autre chose

```
<input list="students"/>
<datalist id="students">
  <option value="Laurent"/>
  <option value="Marc"/>
  <option value="Maurice"/>
</datalist>
```

Soumissions spécifiques

- Création de plusieurs boutons avec des comportement différents :
  - formaction
  - formmethod
  - formnovalidate
  - formtarget

```
<input type="submit" value="Submit">
<input type="submit" formaction="admin.php"
  value="Admin submit">
```

API JAVASCRIPT

API Javascript

- Nouvelles fonctionnalités nécessitant du javascript :
  - Stockage de données
  - Web sockets
  - Geo-localisation
  - Autres

API JAVASCRIPT WEB STORAGE

storage

- Stockage local persistant de données :
  - http://www.w3.org/TR/webstorage/
  - Similaire aux cookies, mais :
    - Pas envoyé dans l'entête HTTP
    - Limite de taille beaucoup plus élevée (5Mo dans la norme)
  - Concurrence et vie privée gérées par le navigateur

- Plusieurs versions :
  - sessionStorage
  - localStorage
  - Base de données (WebSQL, IndexedDB)

storage

- Storage = liste d'éléments stockés via une clé

- Interface générique :
  - getItem(key)
  - setItem(key,value)
  - removeItem(key)
  - clear()
  - length : nombre d'éléments stockés
  - key(n) : accès par indice

localStorage / sessionStorage

- localStorage :
  - Données non supprimées même après fermeture du navigateur
  - Similaire aux cookies dans l'esprit

- Slots :
  - localStorage
  - localStorage.setItem(key, value)
  - localStorage.getItem(key)

- sessionStorage :
  - Idem mais conservation moins garantie

Modification du stockage

- Événement storage :
  - Appelé en cas de changement du storage
    - Dans une autre fenêtre
  - window.addEventListener('storage', function(event){});

- Propriétés accessibles de event :
  - storageArea : objet storage modifié
  - key : clé modifiée
  - oldValue, newValue
  - url : adresse du document modifié

IndexedDB et WebSQL

- Stockage local sous forme de base de données
  - Pour quoi faire ?
- IndexedDB :
  - Stockage hiérarchique clés/valeurs, proche de NoSQL
  - W3C : "A query language can be layered on this API"
- WebSQL = sqlite :
  - SQL classique avec quelques limitations
  - Pas dispo sur tous les navigateurs

WebSQL - exemple

```
var db;
if(window.openDatabase){
  db = openDatabase('ma_base', '1.0', 'database', 2000000);

  db.transaction(function(tx) {
    tx.executeSql("CREATE TABLE Test (id REAL UNIQUE, text TEXT)");
  });

  db.transaction(function(tx) {
    tx.executeSql(
      'INSERT INTO Test (id, text) VALUES (?, ?)',
      [1, "test"]
    );
  });
}
```

Dernière solution

- Mode hors ligne
  - Navigation
  - Sauvegarde
  - Synchro dès qu'on revient en ligne

- Cf manifest

API JAVASCRIPT GEOLOCALISATION

Principe

- API de localisation du navigateur :
  - http://www.w3.org/TR/geolocation-API/
- Via l'objet :
  - navigator.geolocation
- Selon le terminal et le mode de connexion, on utilise :
  - GPS interne
  - Adresse IP ou MAC, RFID
  - Borne 2G/3G
- L'utilisateur est prévenu et doit accepter la geo-localisation

Méthodes

- Déterminer la position actuelle :
  - Calcule la position actuelle puis appelle le callback
  - void getCurrentPosition(fun_ok, fun_ko, options)
    - fun_ko et options ne sont pas obligatoires
- Suivre la position :
  - watchPosition(fun_ok, fun_ko, options)
    - fun_ko et options ne sont pas obligatoires
  - clearPosition(id)

getCurrentPosition

- getCurrentPosition(function_ok, function_ko, options)

- function_ok(position) {}
  - position.coords :
    - latitude, longitude, altitude et heading
    - accuracy et altitudeAccuracy
    - speed, timestamp
  - position.timestamp

getCurrentPosition

- getCurrentPosition(function_ok, function_ko, options)

- function_ko(error) {}
  - error.code :
    - PERMISSION_DENIED=1, POSITION_UNAVAILABLE=2, TIMEOUT=3
  - error.message

- Options :
  - enableHighAccuracy : chercher la meilleure position (lent, couteux)
  - timeout : temps maximum avant appel du callback de succès
  - maximumAge : utilisation d'une position stockée en cache

watchPosition

- long watchPosition(fun_ok, fun_ko, options)
  - Retourne un identifiant de suivi
- clearPosition(id)
  - Annule le suivi de l'identifiant associé
- Premier appel à watchPosition :
  - Retourne un identifiant
  - Exécute la recherche de manière asynchrone
  - Appelle le callback à chaque changement de position (?)

API JAVASCRIPT WEBSOCKETS

Objectif

- Établir une connexion client serveur persistante :
  - Exemple : chat avec mises à jour automatiques chez tous les clients

Objectif

- Établir une connexion client serveur persistante :
  - Exemple : chat avec mises à jour automatiques chez tous les clients
- Solution : créer une connexion "persistante" en Ajax
  - Établissement périodique d'une connexion (Ajax polling)
  - Si données nouvelles : traitement
  - Sinon : (attente puis) création d'une nouvelle connexion
- Couteux pour le serveur, le réseau
  - Rafraichir en fonction du besoin (temps réel, monitoring)

API websocket

- Ouverture et maintien de connexion ouverte
  - http://www.w3.org/TR/websockets/
  - Norme pour le protocole de communication (TCP full duplex)
- Prévoir un fallback si API non disponible :
  - Ajax polling
  - Flash
- Création :
  - var ws = new Websocket(url, [protocoles]);
    - Les protocoles sont optionnels

Création et événements

- Envoi de message :
  - ws.send(message);

- Callbacks
  - ws.onopen = function(evt) {…} :
    - Ouverture de la connexion
  - ws.onmessage = function(evt) {…} :
    - Appelé à chaque réception de message
  - ws.onerror = function(evt) {…} :
    - Traitement des erreurs
  - ws.onclose = function(evt) {…} :
    - Fermeture de la connexion

Coté serveur

- Nécessite un serveur qui comprenne le protocole
- Des implémentations en :
  - C/C++
  - Erlang
  - Java
  - .NET Framework
  - Node.js
  - Perl
  - PHP
  - Python
  - Ruby
- Voir la page wikipedia webSocket pour une liste exhaustive

API Server-sent events

- Uniquement serveur vers client :
  - Correspond à la moitié seulement des webSockets !
  - Utilise HTTP donc pas besoin de serveur spécifique

- Création et événements
  - var source = new EventSource('updates.cgi');
  - source.onmessage = function (e) {alert(e.data); };
  - source.onopen = function (e) {alert(e.data); };
  - source.onerror = function (e) {alert(e.data); };

API JAVASCRIPT QUELQUES AUTRES

Notifications

- Emettre des alertes pour l'utilisateur :
  - http://www.w3.org/TR/notifications/
    - Pas la version implémentée dans Chrome
    - Quasiment pas utilisable dans les autres navigateurs pour l'instant
  - Proche des alert mais moins intrusif

- La demande d'autorisation de notification doit venir de l'utilisateur
  - window.webkitNotifications.requestPermission()

- Création et affichage (version Chrome) :
  - notif = window.webkitNotifications.createNotification('Icone', 'Titre', 'Contenu')
  - notif.ondisplay=function(){}
  - notif.notif.onclose=function() {}
  - notif.show();
  - notif.cancel();

Selector

- Permet de récupérer des éléments de la page :
  - http://www.w3.org/TR/selectors-api/
  - Similaire à $("...") en jQuery
    - Mais moins générique et moins simple à traiter ensuite
- document.querySelector() :
  - Retourne le premier élément vérifiant la sélection
- document.querySelectorAll()
  - Retourne tous les éléments vérifiant la sélection
- Sélection : norme CSS3
  - document.querySelector("#bar, #foo");

Page visibility

- Permet de savoir si la page est visible
  - http://www.w3.org/TR/page-visibility/
- Objet document :
  - document.hidden : statut visible ou caché
  - visibilityChange : événement de changement
  - document.visibilityState : hidden, visible, prerender, unloaded
- Avant la norme :
  - Mozilla : mozHidden, mozvisibilitychange
  - IE : msHidden, msvisibilitychange
  - Chrome : webkitHidden, webkitvisibilitychange

Drag and drop

- Glisser-déposer pour des objets
  - http://www.w3.org/TR/html5/
  - Similaire à jQuery
- Élément déplaçable :
  - Attribut draggable="true"
  - Évenement drag
- Zone d'accueil :
  - Événements dragover et drop
- Drag data store :
  - Stockage et gestion de l'objet déplacé

Autres

- UserMedia : accès aux périphériques locaux
  - Webcam, micro
  - http://www.w3.org/TR/mediacapture-streams/

- Web workers :
  - Création de threads (vrais thread OS)
  - Le code du worker se trouve dans un fichier js distinct

```
var myWorker = new Worker("worker_src.js");
myWorker.onmessage = function (oEvent) {
  console.log("Message from the worker");
};
```

API JAVASCRIPT CANVAS

Canvas

- Balise spécifique pour dessiner en javascript :
  - Nombreuses fonctionnalités classiques
    - Rectangle, cercle, lignes, dégradés, insertion texte et d'image
  - Fonctionnalités avancés :
    - Moteur 3D : webGL
    - http://bruno-simon.com/src/normal/img/articles/three-js/webgl-2.html
  - Plus d'infos la semaine prochaine

- Voir aussi :
  - Création de SVG (balise svg)

Canvas - exemple

```
var canvas = document.getElementById("canvas")
var ctx = canvas.getContext('2d');

// rectangle direct
ctx.fillStyle="grey";
ctx.fillRect(0,0,50,50);
ctx.strokeStyle="black";
ctx.strokeRect(0,0,50,50);

// rectangle avec chemin
ctx.beginPath();
ctx.rect(50, 50, 50, 50);
ctx.fillStyle = 'yellow';
ctx.fill();
```