Cours HTML 5 Avancé

Les APIs HTML5 : Géolocalisation, Local Storage

Voyons maintenant les nouvelles technologies apportées par HTML5 telles que :

- Balises media et contrôle audio/vidéo
- Dessin avec Canvas
- Géolocalisation
- Glisser / Déposer
- Historique
- Local Storage

Les balises media

1. Les principales balises media

1. La balise <audio>

La balise <audio> permet de mettre un fichier son à disposition pour l'écoute. Exemple:

```
<audio src="son.mp3" autoplay="false" controls="">
<a href="son.mp3">Télécharger le fichier son</a>
</audio>
```

2. La balise <video>

La balise <video> permet de mettre un fichier son à disposition pour lecture. Exemple:

```
<video src="video.ogv" controls="true" poster="poster.jpg" autoplay="autoplay"
preload="auto">
<a href="video.ogv">Télécharger le fichier son</a>
</video>
```

Tous les attributs de <video> sont valables pour la balise <audio>, à part l'attribut poster.

3. La balise <source>

La balise <source> permet de proposer différentes alternatives de format au navigateur pour la lecture d'un fichier media.
Exemple pour un fichier son:

```
<audio>
    <source src="son.oga" type="audio/ogg">
    <source src="son.mp3" type="audio/mpeg">
    <source src="son.aac" type="audio/mp4">
    Ce navigateur ne supporte pas l'élément audio
</audio>
```

Exemple pour un fichier vidéo:

```
<video>
  <source src="video.mp4" type="video/mp4">
  <source src="video.3gp" type="video/3gpp" media="handheld">
  <source src="video.ogv" type="video/ogg; codecs=theora, vorbis">
  <source src="video.webm" type="video/webm; codecs=vp8, vorbis">
  Ce navigateur ne supporte pas l'élément video
</video>
```

2. Interface de contrôle et événements

1. Contrôler la lecture

Il est possible de contrôler la lecture à l'aide des fonctions javascript play() et pause().
Pause() sert à la fois à mettre le son en pause et à l'arrêter complètement. Exemple:

```
<video src="video.ogv" id="mavideo">
  <a href="video.ogv">Télécharger la vidéo</a>
</video>
<p>
  <button type="button" onclick="vid1.play();">Play</button>
  <button type="button" onclick="vid1.pause();">Pause</button>
  <button type="button" onclick="vid1.pause();vid1.currentTime=0;">Stop</button>
</p>
```

```
<script>
  var vid1 = document.getElementById("mavideo");
</script>
```

2. Surveiller les événements

La gestion des événements s'effectue en Javascript, afin de déclencher des fonctions de rappel (callback) lorsqu'une action est déclenchée.
Exemple de gestion classique d'événements en javascript:

```
<video src="video.ogv" id="mavideo" controls></video>
<script>
var vid1 = document.getElementById("mavideo");

vid1.onended = function() {
    alert('Fin');
};

vid1.onpause = function() {
    alert('Pause');
};

vid1.onplay = function() {
    alert('Lecture');
};
</script>
```

Exemple de gestion classique d'événements en javascript avec addEventListener() (recommandé):

```
<video src="mavideo.webm" id="mavideo" controls></video>
<script>
var vid1 = document.getElementById('mavideo');
vid1.addEventListener("play", function() {
    alert("Play");
},true);
```

```
vid1.addEventListener("ended", function() {
    alert("Fin");
},true);

vid1.addEventListener("pause", function() {
    alert("Pause.");
},true);
</script>
```

Format de la méthode addEventListener:
element.addEventListener(type, listener, useCapture);

Seuls Firefox et Internet Explorer 9 implémentent actuellement les événements javascript pour la gestion de la vidéo.

3. Détecter les erreurs de lecture

Un gestionnaire d'erreurs est disponible, avec 4 options:

- MEDIA_ERR_ABORTED
- MEDIA_ERR_NETWORK
- MEDIA_ERR_DECODE
- MEDIA_ERR_SRC_NOT_SUPPORTED

Exemple:

```
<script>
document.getElementById("mavideo").onerror = function(event)
{
    switch(event.target.error.code)
    {
        case event.target.error.MEDIA_ERR_ABORTED:
        alert('Lecture annulée.');
        break;
        case event.target.error.MEDIA_ERR_NETWORK:
        alert('Indisponibilité réseau.');
        break;
        case event.target.error.MEDIA_ERR_DECODE:
        alert('Erreur de corruption ou non supporté.');
        break;
```

```
        case event.target.error.MEDIA_ERR_SRC_NOT_SUPPORTED:
        alert('Erreur de chargement ou non suporté.');
        break;
        default:
        alert('Erreur inconnue.');
        break;
    }
};
</script>
```

4. Détection du support avec canPlayType()

La fonction canPlayType() permet également de vérifier si un navigateur prend en charge un type MIME. Elle prend en paramètre optionnel un type MIME et retourne une réponse:
• une chaîne vide : le conteneur vidéo ou son format de compression n'est pas supporté
• maybe : le conteneur est peut-être supporté, mais incertitude sur les codecs
• probably : le conteneur et les codecs sont probablement supportés

Exemple:

```
if (video.canPlayType("video/mpeg")){
 //jouer la video
}
```

L'ordre de priorité de support des différents formats est réparti de la manière suivante:
WebM (VP8/Vorbis) supporté par Google, Apple et Mozilla
Ogg (Theora/Vorbis) pour les anciennes versions de Firefox, Chrome et Opera.
MP4 pour toutes les variantes de Safari
FLV en alternative pour les navigateurs sans support HTML 5

Le dessin avec Canvas

1. L'élément <canvas>

1. Base de départ et contexte graphique

La balise <canvas> permet de définir une zone de tracé accessible en Javascript pour dessiner des formes ou effectuer des manipulations d'images.
Exemple d'initialisation de canvas:

```
<canvas id="dessin" width="640" height="480">
Ce navigateur ne supporte pas canvas
</canvas>
<script>
var moncanvas = document.getElementById('dessin');
if(moncanvas.getContext) {
    var ctx = moncanvas.getContext('2d');
    // D'autres fonctions de dessin
}
```

2. Coordonnées

Les coordonnées sont définies dans un système cartésien et débutent dans le coin supérieur gauche de la zone à (0,0). Toutes les valeurs sont exprimées en pixels. Deux axes sont disponibles : horizontal (x) et vertical (y).

2. Formes géométriques

1. Présentation des fonctions

3 fonctions sont disponibles pour le tracé de rectangles

- fillRect(x,y,w,h) => Rectangle plein
- strokeRect(x,y,w,h) => Rectangle surligné
- clearRect(x,y,w,h) => Rectangle vide (efface)

2. Tracé géométrique simple

Un exemple simple de tracé du drapeau Suisse:

```
<canvas id="dessin" width="640" height="480">
Ce navigateur ne supporte pas canvas
</canvas>
<script>
var moncanvas = document.getElementById('dessin');
if(moncanvas.getContext) {
    var ctx = moncanvas.getContext('2d');
    // Largeur de trait de 2 pixels
```

```
    ctx.lineWidth = 2;
    // Remplissage rouge
    ctx.fillStyle = "red";
    // Contour gruis
    ctx.strokeStyle = "#ccc";
    // Un rectangle plein (rouge)
    ctx.fillRect(10, 10, 200, 100);
    // Deux rectangles effaçant le dessin (blancs)
    ctx.clearRect(100,20, 20, 80);
    ctx.clearRect(70,50, 80, 20);
    // Un dernier rectangle (trait gris) pour le cadre
    ctx.strokeRect(1, 1, 220, 120);
}
</script>
```

3. Chemins

1. beginPath() et closePath()

Pour une plus grande liberté dans le tracé de formes plus complexes, canvas repose sur les chemins. Les chemins sont définis de point en point.
Exemple de gestion de chemin avec beginPath() et closePath():

```
<canvas id="dessin" width="640" height="480">
Ce navigateur ne supporte pas canvas
</canvas>
<script>
    //création du canvas
    var moncanvas = document.getElementById('dessin');
    if(moncanvas.getContext) {
        //initialisation du contexte
        var ctx = moncanvas.getContext('2d');
        //définition d'une couleur remplissage et bordure
        ctx.fillStyle = "#ff9900";
        ctx.strokeStyle = "#990000";
        //1er tracé avec closePath()
        ctx.lineWidth = 20;
        ctx.beginPath();
```

```
        ctx.moveTo(25,25);
        ctx.lineTo(100,100);
        ctx.lineTo(25,100);
        ctx.closePath();
        //exécution du remplissage
        ctx.fill();
        //exécution du tracé
        ctx.stroke();

        //deuxième tracé sans closePath()
        ctx.beginPath();
        ctx.moveTo(300,100);
        ctx.lineTo(220,100);
        ctx.lineTo(220,25);
        ctx.lineTo(300,100);
        //exécution du remplissage
        ctx.fill();
        //exécution du tracé
        ctx.stroke();
}
</script>
```

2. moveTo() et lineTo()

La méthode moveTo() positionne le point de départ du tracé.
La méthode lineTo() positionne le point d'arrivée du tracé.
Exemple de tracé de maison:

```
<canvas id="dessin" width="640" height="480">
Ce navigateur ne supporte pas canvas
</canvas>
<script>
    //création du canvas
    var moncanvas = document.getElementById('dessin');
    if(moncanvas.getContext) {
        //initialisation du contexte
        var ctx = moncanvas.getContext('2d');
        // Toit
```

```
        ctx.moveTo(40, 80);
        ctx.lineTo(80, 40);
        ctx.lineTo(120, 80);
        ctx.stroke();
        // Murs
        ctx.moveTo(60, 80);
        ctx.lineTo(60, 120);
        ctx.lineTo(100, 120);
        ctx.lineTo(100, 80);
        ctx.stroke();
    }
</script>
```

3. fill() et stroke()

Ces deux méthodes permettent de gérer respectivement le contour et le remplissage d'une forme. (Cf exemple point 1 beginPath/closePath)

4. rect()

Cette méthode permet de gérer le tracé d'un rectangle simple. Exemple d'ajout d'une porte au tracé précédent:

```
// Porte
ctx.rect(75,100,10,20);
ctx.stroke();
```

5. arcTo()

La méthode arcTo() permet le tracé de courbes.
Définition de la fonction:
arcTo(x1,y1,x2,y2,r) – coord départ, coord arrivée, rayon

Exemple de tracé de courbe:

```
//exemple de tracé de courbe
ctx.beginPath();
ctx.strokeStyle = "LimeGreen";
ctx.moveTo(20,120);
ctx.arcTo(20,30,160,120,20);
```

```
ctx.stroke();
```

6. arc()

La méthode arc() n'est pas paramétrée par des coordonnées, mais par un centre, un rayon et un angle de départ et de fin. Cette méthode agit comme un tracé de compas.
Définition de la fonction:
arc(x,y,r,a1,a2,c) –
coord départ, rayon, angle départ, angle arrivée, sens rotation

Exemple de tracé de cercle:

```
// Tracé de cercle
ctx.fillStyle = "yellow";
ctx.strokeStyle = "orange";
ctx.beginPath();
ctx.arc(150,40,30,0,Math.PI*2,true);
ctx.fill();
ctx.stroke();
```

7. bezierCurveTo()

La méthode bezierCurveTo() permet de tracer des courbes de Bézier. Elle prend en paramètre les corrdonnées d'un point de départ, d'un point d'arrivée et d'une direction.
Définition de la fonction:
bezierCurveTo(x1,y1,x2,y2,x,y) – coord départ, coord arrivée, coord direction

Exemple de courbe de Bézier:

```
ctx.strokeStyle = "turquoise";
ctx.beginPath();
//tracé 1
ctx.moveTo(20, 150);
ctx.bezierCurveTo(80,130,80,180,140,150);
//tracé 2
ctx.moveTo(20, 170);
ctx.bezierCurveTo(80,150,80,200,140,170);
//exécution
ctx.stroke();
```

8. quadraticCurveTo()

La méthode quadraticCurveTo() est une courbe de Bézier qui ne prend pas de direction en paramètre.
Définition de la fonction:
quadraticCurveTo(x1,y1,x2,y2) – coord départ, coord arrivée

Exemple de courbe quadratique:

```
//tracé de courbe quadratique
ctx.beginPath();
ctx.strokeStyle = "orange";
ctx.moveTo(130, 40);
ctx.quadraticCurveTo(150,70,170,40);
ctx.stroke();
```

9. Styles de traits, remplissages et couleurs

Le canvas peut être personnalisé à l'aide des propriétés suivantes:

- fillStyle - remplissage des formes - code couleur CSS, dégradé ou motif
- strokeStyle - lignes autour des formes - code couleur CSS, dégradé ou motif
- lineWidth - largeur de ligne - nombre positif
- lineJoin - style de jointure des lignes - bevel, round, ou miter
- lineCap - forme de fin de ligne butt, round, ou square
- globalAlpha - Transparence générale - nombre positif ou nul (0 à 1)

10. Dégradés

Un dégradé linéaire est créé avec createLinearGradient().
Les couleurs sont modulées progressivement du point de départ au point d'arrivée, sauf si des points intermédiaires sont ajoutés.

Un dégradé radial est créé avec createRadialGradient().
Les trois premiers arguments créent un cercle pour le point de départ, et les trois derniers pour le point de fin. La méthode addColorStop() ajoute des points d'arrêt intermédiaires dans le dégradé.

En général, le premier point de couleur est défini pour 0, et le dernier pour 1. Entre ces deux valeurs, d'autres points peuvent être ajoutés pour moduler le dégradé et ajouter des couleurs intermédiaires (par exemple 0,1 pour 10 %, 0,5 pour 50 %, etc.).

Une fois le dégradé préparé, il peut être appliqué aux propriétés fillStyle ou strokeStyle pour affecter les remplissages et les contours.

Exemple de tracés dégradés linear et radial:

```
//objet de dégradé linéaire
var degradeLineaire = ctx.createLinearGradient(0, 0, 200, 0);
degradeLineaire.addColorStop(0,'limegreen');
degradeLineaire.addColorStop(0.5,'gold');
degradeLineaire.addColorStop(1,'orange');
ctx.fillStyle = degradeLineaire;
ctx.fillRect(0, 0, 600, 100);

//objet de dégradé radial
var degradeRadial = ctx.createRadialGradient(300,250,0, 300,250,300);
degradeRadial.addColorStop(0,'red');
degradeRadial.addColorStop(0.5,'#000099');
degradeRadial.addColorStop(1,'brown');
ctx.fillStyle = degradeRadial;
ctx.fillRect(0, 110, 600, 100);
```

4. Transformations et états du contexte

1. Transformations

Il est possible d'effectuer des transformations d'échelle, de translation et de rotation sur une forme dessinée avec canvas.
Les fonctions disponibles pour la transformation:
scale(x,y) - Modification de l'échelle - coefficients appliqués à x et y
rotate(angle) - Rotation : angle de rotation en radians
translate(x,y) - Translation : déplacements appliqués à x et y

Exemples de transformations:

```
// Étirement
ctx.scale(2,0.5); //coefficients x, y
// Réduction
ctx.scale(0.3,0.5); //coefficients x, y
// Translation
ctx.translate(50,0); //valeurs x et y
```

2. La fonction save()

La fonction save() permet d'enregistrer l'état du canevas à un moment donné.
L'état est sauvegardé dans une pile selon la méthode LIFO (Last In First Out).
Exemple de sauvegarde d'état:

```
// Sauvegarde canevas
ctx.save();
```

3. La fonction restore()

La fonction restore() permet de restaurer un état du canevas préalablement sauvegardé.
L'état est restauré à partir de la pile selon la méthode LIFO (Last In First Out).
Exemple de restauration d'état:

```
// Sauvegarde initiale
ctx.save();
// transformation
ctx.scale(2,2);
// Tous les tracés suivants sont agrandis
ctx.restore();
// Tous les tracés suivants ne sont plus affectés
```

4. Images

Le canvas met à disposition des proprpiétés pour afficher et/ou redimensionner des images jpg, png et gif.
Exemple d'affichage et de redimensionnement:

```
//initialisation du contexte
var ctx = moncanvas.getContext('2d');
// Nouvel objet Image
var img = new Image();
// Définition de la source
img.src = 'planete.png';
// Gestionnaire d’événement load
```

```
img.onload = function() {
    // Dessin de l'image
    ctx.drawImage(img,0,0);
    // Dessin de l'image redimensionnée
    ctx.drawImage(this,450,150,100,100); // objet image,x,y,w,h
}
```

5. Motifs

L'élément canvas permet également de créer des motifs à partir d'une image, à partir des mêmes valeurs que celles utilisées pour la propriété background-repeat en CSS (repeat, repeat-x, repeat-y).
Exemple de création de motif:

```
//création du canvas
var moncanvas = document.getElementById('dessin');
if(moncanvas.getContext) {
    //initialisation du contexte
    var ctx = moncanvas.getContext('2d');
    // Nouvel objet Image
    var img = new Image();
    // Définition de la source
    img.src = "icon.png";
    // Gestionnaire d’événement load
    img.onload = function(){
        // Création du motif à partir de l’image
        // Mêmes valeurs que background-repeat en CSS
        var motif = ctx.createPattern(this, "repeat");
        // Dessin d’un rectangle plein avec ce motif
        ctx.rect(10, 10, 400, 500);
        ctx.fillStyle = motif;
        ctx.fill();
    };
```

5. Texte

1. Propriétés

Les propriétés suivantes sont disponibles pour la gestion du texte:

font - style de police - syntaxe CSS (valeur par défaut : 10px sans-serif)
textAlign - alignement horizontal start (défaut), end, left, center, right
textBaseline - alignement vertical top, hanging, middle, alphabetic (défaut), ideographic, bottom

2. Méthodes

Les méthodes suivantes sont disponibles pour la gestion du texte:

fillText(txt,x,y,maxw) : remplissage - texte, coords, largeur max (optionnel)
strokeText(txt,x,y,maxw) : contour- texte, coords, largeur max (optionnel)
measureText(txt) : Renvoie l'espace nécessaire pour un texte (en pixels)

Exemple de tracé de texte:

```
// Première ligne
ctx.font = '30pt Comic Sans MS';
ctx.textAlign = "start";
ctx.textBaseline = "top";
// Tracé du texte 1
ctx.strokeText("Tracé du texte 1",0,0);
// Deuxième ligne
ctx.font = '50pt Comic Sans MS';
ctx.lineWidth = 3;
ctx.strokeText("Tracé du texte 2",0,25);
// Tracé du texte 2
ctx.font = '20pt Georgia';
ctx.textAlign = "right";
ctx.fillText("...disait la tortue au tatou",350,110);
```

3. Exemple de gestion du texte

Voici 3 exemples différents de tracé de texte :

```
// Tracé du texte 1
ctx.font = '30pt Comic Sans MS';
ctx.textAlign = "start";
ctx.textBaseline = "top";
// Première ligne
ctx.strokeText("Ton thé t'a-t-il",0,0);
// Deuxième ligne
ctx.font = '50pt Comic Sans MS';
ctx.lineWidth = 3;
```

```
ctx.strokeText("ôté ta toux ?",0,25);
// Tracé du texte 2
ctx.font = '20pt Georgia';
ctx.textAlign = "right";
ctx.fillText("...disait la tortue au tatou",350,110);
```

6. Ombres, transparence, composition

1. Propriétés

La gestion des ombres est également possible avec l'élément canevas, à partir des propriétés suivantes:

shadowOffsetX - Étendue de l'ombrage sur l'axe horizontal
shadowOffsetY - Étendue de l'ombrage sur l'axe vertical
shadowBlur - Valeur du flou
shadowColor - Couleur de l'ombre

2. Exemple de gestion d'ombres

```
// Configuration des ombres pour un rectangle
ctx.shadowOffsetX = 0;
ctx.shadowOffsetY = 0;
ctx.shadowBlur = 15;
ctx.shadowColor = 'orange';
// Tracé d'un rectangle
ctx.fillStyle = '#fff';
ctx.fillRect(10,10,150,150);
// Tracé d'un autre rectangle
ctx.fillStyle = '#9f0c9d';
ctx.fillRect(50,50,70,70);
```

3. Transparence

La transparence se gère à partir de la propriété globalAlpha de canvas. La valeur doit être comprise entre 0 et 1.

Exemple de gestion de la transparence:

```
var w = ctx.canvas.width;
var h = ctx.canvas.height;
ctx.fillStyle = "#789";
ctx.globalAlpha = 0.5;
// Tracé de rectangles
var a = Math.random()*(w-60)/2;
var b = Math.random()*(h-60)/2;
for (i=0;i<6;i++) {
    a+=10;
    b+=10;
    ctx.fillRect(a,b,a,b);
}
```

4. Composition

La composition permet de gérer la superposition et/ou la combinaison de remplissages de formes.
La propriété globalCompositeOperation régit la façon dont sont menées les opérations de composition sur le canvas.
Propriétés disponibles:

source-over

La source se retrouve par-dessus la destination (par défaut).

source-in

La source est visible uniquement là où source et destination se recouvrent.
Tout le reste est rendu transparent.

source-out

La source est visible uniquement là où elle ne recouvre pas la destination.

source-atop

La source est visible uniquement là où source et destination se recouvrent.

destination-over

La source est dessinée derrière le contenu existant.

destination-in

Le contenu existant reste visible uniquement là où source et destination se recouvrent. Tout le reste est rendu transparent.

destination-out

Le contenu existant reste visible uniquement là où il ne recouvre pas la source.

destination-atop
Le contenu existant reste visible uniquement là où source et destination se recouvrent. La source est dessinée derrière le contenu existant.

lighter
La couleur est déterminée par addition là où source et destination se recouvrent.

darker
La couleur est déterminée par soustraction là où source et destination se recouvrent.

copy
Seule la source est dessinée, tout le reste est retiré.

xor
Source et destination sont rendues transparentes là où elles se recouvrent, et dessinées de façon normale sinon.

Exemple de gestion de la superposition:

```
//tracé d'un rectangle
ctx.fillStyle = "orange";
ctx.fillRect(10,10,80,80);
ctx.globalCompositeOperation = "source-atop";
ctx.fillStyle = "red"
ctx.beginPath();
//tracé d'un cercle
ctx.arc(100,100,60,0,Math.PI*2,true);
ctx.fill();
```

7. Contrôle clavier et souris

1. Souris

Le contrôle à la souris est disponible avec les événements suivants:
click, dblclick, mousedown, mouseup, mousemove, mouseenter, mouseleave.
Exemple de gestion d'un clic sur le canvas:

```
// Bouton de souris activé
ctx.onmousedown = function(e) {
    alert('canevas cliqué');
};
```

2. Clavier

La gestion du clavier fonctionne avec l'interception de l'événement keypress, keyup ou keydown. La valeur de la touche enfoncée correspond à un code numérique stocké dans la propriété event.keyCode.
Exemple de gestion d'un déplacement à partir du clavier:

```
//ajout d'un écouteur d'événement avec callback
ctx.onkeydown = deplacer;

//fonction de callback
function deplacer(event) {
    switch(event.keyCode) {
        case 38: // Haut
        event.preventDefault();
        //action a effectuer
        break;


        case 40: // Bas
        event.preventDefault();
        //action a effectuer
        break;

        case 39: // Droite
        event.preventDefault();
        //action a effectuer
        break;

        case 37: // Gauche
        event.preventDefault();
        //action a effectuer
        break;
    }
}
```

Géolocalisation

1. Déclencher la localisation

1. Principe

La géolocalisation repose sur les coordonnées de latitude, de longitude et d'altitude.
Lors de l'appel aux fonctions de géolocalisation, un avertissement est affiché par le navigateur, précisant quel site demande l'accès aux informations ainsi que les choix disponibles : accepter ou refuser.

Durant ce temps, la page web n'a accès à aucune des données demandées.

Le code JavaScript et DOM doit être déclaré dans un élément <script>. La géolocalisation fait appel à l'objet geolocation, membre de navigator (ou window.navigator).

Exemple de test de détection de l'API:

```
if(navigator.geolocation) {
    // L'API est disponible
} else {
    // Proposer une alternative
}
```

2. Méthodes disponibles

Deux méthodes de l'objet geolocation sont disponibles:
• getCurrentPosition() pour une requête unique ;
• watchPosition() pour un suivi continu de la position.

Ces méthodes acceptent en arguments :
• une fonction de callback (obligatoire),
• une deuxième fonction de callback (facultative), appelée en cas d'erreur ;
• un ensemble d'options PositionOptions (facultatif).

Exemple avec de géolocalisation avec getCurrentPosition :

```
function affichePosition(position) {
  // Afficher la position dans la page ou sur une carte...
}
navigator.geolocation.getCurrentPosition(affichePosition);
```

Exemple avec de géolocalisation avec watchPosition :

```
function surveillePosition(position) {
  // Rafraîchir régulièrement la position affichée
  // dans la page ou sur une carte...
}
var survId = navigator.geolocation.watchPosition(surveillePosition);
  // Pour annuler la surveillance continue
function annuler() {
    navigator.geolocation.clearWatch(survId);
}
```

Travailler avec la position et les coordonnées

Un objet de type Position contient des propriétés accessibles en lecture seule :

- timestamp : la date à laquelle la position a été acquise (millisecondes)
- coords : les coordonnées.

La propriété coords de type Coordinates contient des précisions :

- latitude : la latitude (degrés décimaux)
- longitude : la longitude (degrés décimaux)
- accuracy : le coefficient de précision (mètres)
- altitude : l'altitude (optionnel, mètres)
- altitudeAccuracy : la précision sur l'altitude (optionnel, mètres)
- heading : la direction (optionnel, degrés)
- speed : la vitesse (optionnel, mètres/seconde)

Exemple d'affichage de la géolocalisation sur une page:

```
<!-- Un élément HTML pour recueillir les informations -->
<div id="maposition"></div>
<script>
// Fonction de callback en cas de succès
function succesGeo(position) {
    var infopos = "Position déterminée : <br>";
    infopos += "Latitude : "+position.coords.latitude +"<br>";
    infopos += "Longitude: "+position.coords.longitude+"<br>";
    infopos += "Altitude : "+position.coords.altitude +"<br>";
    document.getElementById("maposition").innerHTML = infopos;
}
// Demande de position
navigator.geolocation.getCurrentPosition(succesGeo);
</script>
```

3. Gestion des erreurs

Les méthodes getCurrentPosition() et watchPosition() acceptent une fonction de callback pour la gestion des erreurs en deuxième argument.

Cette fonction reçoit un objet PositionError qui renseigne sur la cause de l'erreur en cas d'insuccès, à l'aide de deux propriétés: code et message.

La proriété code peux contenir les valeurs suivantes:
• UNKNOWN_ERROR : La localisation a échoué pour une raison inconnue.
• PERMISSION_DENIED : La localisation a échoué (refus utilisateur).
• POSITION_UNAVAILABLE : La position n'a pu être déterminée
• TIMEOUT : Le temps imparti est écoulé sans qu'une position ne soit obtenue.

4. Options supplémentaires

Les méthodes getCurrentPosition() et watchPosition() acceptent un objet PositionOptions en troisième argument (facultatif).

enableHighAccuracy : demande de fournir une position en haute précision, si le matériel le supporte et que l'utilisateur en donne la permission.

timeout : spécifie explicitement un délai de temps d'attente au bout duquel la fonction retournera une erreur (voir code précédent) si la position n'a pu être

déterminée.

maximumAge : autorise le système à répondre immédiatement avec une valeur mémorisée en cache, provenant d'une requête précédente.

Exemple complet de géolocalisation avec gestion des erreurs:
http://codepen.io/smoothie-creative/pen/77126f0b45725d41807a6e6907e41f72

Local Storage

Le local storage permet de stocker des informations localement, de manière plus poussée que les cookies, afin que l'internaute puisse travailler de son navigateur même non connecté. Il est ensuite possible de récupérer les informations une fois la connexion établie et les enregistrer sur le serveur à distance.

Tuto :
http://www.sitepoint.com/html5-web-storage/

Pushstate

Manipuler l'historique du navigateur https://developer.mozilla.org/en-US/docs/Web/Guide/API/DOM/Manipulating_the_browser_history

Sert par exemple dans une application dynamique où il n'y a pas de rechargement de page à chaque action. Le pushstate permet de revenir en arrière sur la dernière action, au lieu de revenir sur la dernière page.

Exemple du menu sur wpinalps.com