Formulaire PHP/Javascript/AJAX

PHP

Code de base

```
<?php
//Un commentaire
echo 'Mon texte';
?>
```

Conditions

```
if(condition) {
	//Faire qqchose
} else if(autreCondition) {
	//Faire autre chose
} else {
	//Faire encore autre chose
}
```

ou

```
switch($var) {
	case x:
		//Faire qqchose
		break;
	default:
		//Sinon faire autre chose
		break;
}
```

Boucles

```
while(condition d'arrêt) {
	//Instructions à répéter
}

do {
	//Instructions à répéter
} while(condition d'arrêt);


for($i = 0 ; condition d'arrêt ; $i++) {
	//Instructions à répéter
}

foreach($tableau as $key) {
	//Instructions à répéter
}
```

Structures de langage

- `echo` : affiche qqchose (une variable, une chaîne de caractères, un nombre...)
- `include` : inclut un fichier dans le fichier courant
- `$var` : une variable
- `$var['index']` : case d'un tableau
- Déclaration fonction :

```
function maFonction($param) {
   //Faire qqchose
   return $resultat;
}
```

- `maFonction($param);` : utilisation
- $_GET, $_POST : variables tableau pour récupérer les paramètres d'url, de formulaire

Javascript

Étapes

- On écrit le js dans un fichier à part (de préférence)
- On ajoute un appel à js dans html, via OnClick ou autre.

Conditions, boucles, fonctions

Idem que PHP, sauf `foreach` qui n'existe pas.

Structures de langage

- `alert(var);` : affiche le contenu d'une var dans une popup.
- `confirm(var);` : idem, mais propose ok/annuler.
- `prompt(var);` : idem, mais avec un champ texte.
- `maVariable` : une variable, à déclarer avec `var`

DOM

À préfixer avec le domaine de recherche (ex : document.)

- `getElementById('id');`
- `getElementsByClassName('class');`
- `getElementsByTagName('balise');`
- `getElementsByName('nom');`
- `appendChild(elementDOM);`
- `removeChild(elementDOM);`
- `createElement('tagName');`
- `createTextNode('texte');`
- `setAttribute('attribut', valeur);`
- `innerHTML` : contenu d'une balise

AJAX

Appel de PHP via Javascript. Démarche :

```
var xhr = getXhr();
var requete = 'param1=value1&param2=value2';
xhr.open('requete', fichier.php);
//Uniquement pour POST
xhr.setRequestHeader("Content-Type",
"application/x-www-form-urlencoded");
xhr.onreadystatechange = function() {
	if (xhr.readyState == 4 && xhr.status
	== 200) { //Traitement sans erreur
		alert(xhr.responseText);
	} else if (xhr.readyState == 4 &&
	xhr.status != 200) { // En cas
	d'erreur
	   alert('Erreur');
	}
}
xhr.send(requete);
```

Méthodes disponibles : GET ou POST
Formats de réponse : responseText, responseXML. Préciser le format aussi dans PHP avec `header('Content-type: text/xml');` dans ce dernier cas.

Formulaire HTML/CSS

HTML

Page de base

```
< !DOCTYPE html>
<html lang="fr">
        <head>
                <meta charset="utf-8" />
                <title>Ma page</title>
        </head>
        <body>
                Mon contenu
        </body>
</html>
```

Balises à connaître

- <link rel="stylesheet" type="text/css" href="url" /> : inclure une feuile de style
- : paragraphe
-
 : retour à la ligne
- <hx></hx> : titre (avec $1 \leq x \leq 6$)
- : conteneur neutre de type block
- : conteneur neutre de type inline
- : image (src :url, alt : texte de remplacement)
- <form method="methode" action="page.php"></form> : conteneur de formulaire (method : post ou get, action : nom de la page de traitement en php – peut être la même que la page courante)
- <input type="type" name="nom" /> : champ de formulaire (type : text, password, reset, submit,...) et name un identifiant pour pouvoir retrouver le contenu en php.
- <textarea cols="nbDeColonnes" rows="nbDeLignes" name="nom"></textarea> : champ de texte multi-ligne.
- <fieldset><legend>Titre</legend></fieldset> : sous-conteneur de formulaire.

Inline et block

Deux comportements pour les balises :

- **inline** : comportement en ligne, pas de saut de ligne après la balise. (ex : <img /> ou)
- **block :** comportement bloc, saut de ligne après la balise. (ex :)

Attributs

Une balise peut avoir un ou plusieurs attributs. On les place après le nom de la balise (ex:).
Parmi ces attributs, deux qui serviront beaucoup en javascript et en CSS :

- id : donne un identifiant **unique** à la balise
- class : donne un identifiant **commun** à plusieurs balises

Et en PHP : name permettra de récupérer le contenu d'un champ d'un formulaire.

CSS

Syntaxe de base

```
Balise {
        propriété:valeurs;
}
```

Balise

Utiliser le nom de la balise directement, ou cibler un élément par son id (préfixe #) ou sa class (préfixe .)
On peut enchaîner les éléments pour établir une parenté (ex : **form input** ciblera tous les champs contenus dans un formulaire).
Plusieurs éléments peuvent avoir le même style : on les sépare avec des virgules.

Propriétés à connaître

Mise en forme du texte

Propriété	Valeurs possibles	Description
text-align	left, right, center, justify	Aligne le texte
font-size	Taille en pixels, en %, en em...	Taille du texte
font-weight	normal, bold	Gras ou non
font-style	italic, normal	Italique ou non

Couleur et fond

color	Code hexa, rgb ou nom anglais pour certaines	Couleur du texte
background-color	idem	Couleur de fond
background-image	url('lienImage')	Image de fond

Boîtes

height	Taille en px, ...	Hauteur boîte
width	Idem	Largeur boîte
padding-left, ...	Idem	Marge intérieure
margin-left, ...	Idem	Marge extérieure
border	Épaisseur, type (solid, dotted, dashed...), couleur du trait	Bordure

Positionnement

float	left, right	Flottant
clear	Left, right, both	Annule un flottant
position	absolute, relative	Positionne un élément
top, right, bottom, left	Position	Coordonnées du positionnement