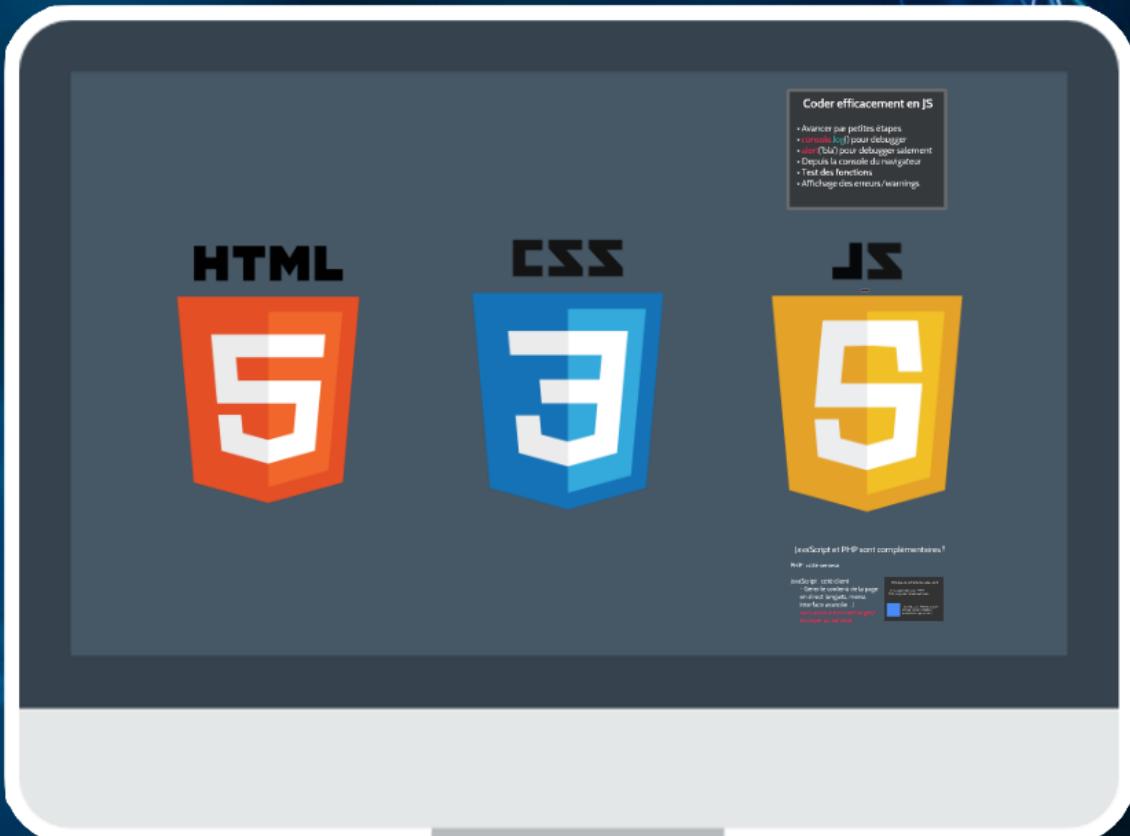






Client





Prezi

JavaScript et PHP sont complémentaires !

PHP : côté serveur

JavaScript : côté client

- Gérer le contenu de la page en direct (onglets, menu, interface avancée ...)

**sans avoir à tout recharger/
envoyer au serveur.**

Principe du dynamisme côté client

Le navigateur télécharge : HTML, CSS, images des scripts exécutables



Les scripts sont interprétés pour :
interagir avec le navigateur
manipuler la page courante

Principe du dynamisme côté client

Le navigateur télécharge : HTML,
CSS, images des scripts exécutables



Les scripts sont interprétés pour:
interagir avec le navigateur
manipuler la page courante



Prezi

Coder efficacement en JS

- Avancer par petites étapes
- `console.log()` pour debugger
- `alert('bla')` pour debugger salement
- Depuis la console du navigateur
- **Test des fonctions**
- Affichage des erreurs/warnings



JS

KEEP CALM AND LEARN JAVASCRIPT

JS + HTML

```
<html>
<head>
<script src="js/monFichier.js"></script>
</head>
<body> ... </body>
</html>
```

Le script JS est un fichier indépendant
Séparation des préoccupations
des bibliothèques JS ré-utilisables

HTML+JS dans la pratique

[Programmation événementielle] [Programmation asynchrone]
Déclencher un événement à l'aide d'un document
Événements :
- Utiliser le code chargé
- Utiliser les fonctions
- Déclarer une variable
- Utiliser les méthodes

!/ JavaScript != Java

Langage multi-paradigmes:
• procédural
• orienté objet
• fonctionnel

Comme pour PHP : DE LA RIGUEUR !!
ambiguités
typage dynamique
typage faible (conversions implicites) très
laxistes sur l'absence de variables
de fonctions

Les bases : un langage de programmation complet



Le DOM Document Object Model

- Modèle arborescent représentant la page
- Tous les noeuds de l'arbre sont des éléments du document



$/\!\backslash$ JavaScript != Java

Langage multi-paradigmes:

- procédural
- orienté objet
- fonctionnel

Comme pour PHP : DE LA RIGUEUR !!
ambiguités
typage dynamique
typage faible (conversions implicites) très
laxistes sur l'absence de variables
de fonctions



Les bases : un language de programmation complet

les types de base

nombres	12;7.48;NaN; -Infinity; Infinity
strings	"toto","tata'
booleen	false, true
objet	fonctions, tableaux, dates, regex, erreurs
indéfini	undefined (absence d'initialisation) null (absence délibérée)

```
var counter; // Une variable globale
function add(x,y){ // test locale
    return x+y;
}
function factorel(n){
    if(n == 0)
        return 1;
    else
        return n * factorel(n-1);
}
```

Les fonctions

```
function foo(bar, baz) {
    return bar + baz;
}
```

- foo(2, 3) retourne 5
- foo(2,3,4,false) retourne 5
(paramètres supplémentaires ignorés)
- foo(2) retourne NaN
(paramètre manquant == undefined)

!\\

null != undefined 47 + undefined == NaN 47 + null == 47

Les objets

Des paires clef -> valeur

```
//Définition
var o1 = new Object();
var o = {};
var o = {clef1: 1, 'clef2': false};
```

//lecture

```
var v1 = o.clef1
var v2 = o['clef1']
```

cf PHP

```
//écriture
o.clef1 = v;
o['clef'] = v;
```

Valeurs hétérogènes et clefs non consécutive autorisées

Parcours d'un tableau

Attention aux trous !

```
function myLoop(arr, index) {
    if(index < 0 || index > arr.length - 1) {
        return;
    }
    var v = arr[index];
    arr[index] = undefined; // Marquer la valeur comme trouée en php
    myLoop(arr, index + 1); // Passer l'index + 1 comme trouée en php
    arr[index] = v;
}
```

Objets Perso

Définition et création

```
new
this
creation une instance:
la fonction appelle devient le constructeur
référence l'instance de l'objet et ses méthodes
function Personne(nom) {
    this.nom = nom;
    this.nom = nom;
    this.nomComplete = function() {
        return this.nom + " " + this.nom;
    }
}

var p = new Personne("Wyle");
p.nomComplete(); // => Wyle Wyle
```

Tableaux

Un objet pour des clés entières + propriété length prédefinie

```
//définition
var a1 = new Array();
var a3 = [];
var a3 = [eric,"butters","kenny"];
```

//lecture //écriture
var v = t[0]; t[1] = 'dr. chaos';

// **length** == index du prochain element
t[0] = 'the coon'; t[10] = 'captain obvious'

t.length == 11

fonctions prédefinies:

ajout/retrait : pop, push, shift, unshift
manipulations usuelles : reverse, sort, slice



les types de base

nombres 12;7.48;NaN; -Infinity; Infinity

strings "toto","tata"

booleen false, true

objet fonctions, tableaux, dates, regex, erreurs

indéfini undefined (absence d'initialisation) null
(absence délibérée)

Les fonctions

```
function foo(bar, baz) {  
    return bar + baz;  
}
```

- `foo(2, 3)` retourne 5
- `foo(2,3,4,false)` retourne 5
(paramètres supplémentaires ignorés)
- `foo(2)` retourne NaN
(paramètre manquant == undefined)

/!\

`null != undefined`

`47 + undefined == NaN`

`47 + null == 47`

```
var counter; // Une variable globale
function add(x, y) {
  var r = x + y; // r est locale
  return r;
}

function inc() {
  counter++;
}
```

```
/* Calcule un factoriel de manière recursive */
function factoRec(n) {
  if (n == 0) {
    return 1;
  }
  return n * factoRec(n - 1);
}
```

Les objets

Des paires clef -> valeur

cf PHP

//Définition

```
var o1 = new Object();
```

```
var o = {};
```

```
var o = {'clef1': 1, 'clef2': false};
```

//lecture

```
var v1 = o.clef1
```

```
var v2 = o['clef1']
```

//écriture

```
o.clef1 = v;
```

```
o['clef'] = v;
```

Valeurs hétérogènes et clefs non consécutives autorisées

Tableaux

Un objet pour des clés entières + propriété length prédéfinie

//définition

```
var a1 = new Array();
```

```
var a3 = [];
```

```
var a3 = ['eric','butters','kenny'];
```

//lecture

```
var v = t[0];
```

//écriture

```
t[1] = 'dr. chaos';
```

// **!** length == index du prochain élément

```
t[0] = 'the coon'; t[10] = 'captain obvious'
```

t.length == 11

fonctions prédéfinies:

ajout/retrait : pop, push, shift, unshift

manipulations usuelles : reverse, sort, slice

Parcours d'un tableau

Attention aux trous !

```
function avg_0_en_JavaScript(notices) {  
    var s = 0;  
    for (var i = 0; i < notices.length; i++) {  
        s += notices[i];  
    }  
    return s / notices.length;  
}  
  
function avg_bien(notices) {  
    var s = 0;  
    var nb = 0;  
    for (var i in notices) { // parcours les index - comme foreach en php  
        s += notices[i];  
        nb++;  
    }  
    return s / nb;  
}
```

Objets Perso

Définition et création

new

création une instance.

la fonction appelée devient le constructeur

this

référence l'instance de l'objet et ses méthodes

```
function Etudiant(p, n) {  
    this.prenom = p;  
    this.nom = n;  
    this.nomComplet = function() {  
        return this.prenom + ' ' + this.nom  
    }  
}
```

```
var b = new Etudiant('Kyle', 'Browslovsky');  
b.nomComplet(); // == 'Kyle Browslovsky'
```

JS + HTML

```
<html>
  <head>
    <script src='js/monFichier.js'></script>
  </head>
  <body> ... </body>
</html>
```

Le script JS est un fichier indépendant
Séparation des préoccupations
des bibliothèques JS ré-utilisables

Objets JS prédéfinis

document	la page HTML actuelle
navigator	le navigateur utilisé
screen	l'écran de l'internaute
history	historique
location	url de la page courante

Pour chercher/modifier des balises XHTML, des propriétés CSS, changer d'URL, exécuter des requêtes, ...

Objets JS prédéfinis

document

la page HTML actuelle

navigator

le navigateur utilisé

screen

l'écran de l'internaute

history

historique

location

url de la page courante

Pour chercher/modifier des balises XHTML, des propriétés CSS, changer d'URL, executer des requêtes, ...



HTML+JS dans la pratique

Programmation évènementielle :
l'interpréteur JS va exécuter du code en fonction d'évènement

Exemples d'évènements :

- La balise `<p>` a été chargée
- Toutes les 3 secondes
- Clique sur le bouton
- Passage de la souris sur un élément

`onXXXX="du code javascript"`

```
<body onload="alert('chargé')">
<h1 onclick="alert('hop')">Titre</h1>
<p onmouseover="alert('dessus')">
blablabla
</p>
<p><a href='#' onclick="alert('hop')">
un lien
</a></p>
</body>
```

http://www.w3schools.com/jsref/dom_obj_event.asp

Programmation évènementielle : l'interpréteur JS va exécuter du code en fonction d'évènement

Exemples d'évènements :

- La balise `<p>` a été chargée
- Toutes les 3 secondes
- Clique sur le bouton
- Passage de la souris sur un élément

onXXXX="du code javascript "

```
<body onload="alert('chargé')">
  <h1 onclick="alert('hop')">Titre</h1>
  <p onmouseover="alert('dessus')">
    blablabla
  </p>
  <p><a href="#" onclick="alert('hop')">
    un lien
  </a></p>
</body>
```

http://www.w3schools.com/jsref/dom_obj_event.asp

Le DOM

Document Object Model

- Modèle arborescent représentant la page
- Tous les noeuds de l'arbre sont des éléments du document

Des méthodes pour naviguer dans le DOM:

```
var x = document.getElementById('list')
x.childNodes;
x.getElementsByTagName('li')[0]
...
```

http://www.w3schools.com/js/js_htmldom.asp

<http://users.polytech.unice.fr/~hermenie/adw/cours/base-js/dom.html>

Des méthodes pour modifier le DOM:

```
var x = document.getElementById('list')
x.style.color = 'blue';
var items = document.getElementsByTagName('li')
items[0].innerHTML = 'hello'
x.innerHTML += '<li>Nouvel item</li>'
...
```

Des méthodes pour naviguer dans le DOM:

```
var x = document.getElementById('list')
x.childNodes;
x.getElementsByTagName('li')[0]
```

...

http://www.w3schools.com/js/js_htmldom.asp

<http://users.polytech.unice.fr/~hermenie/adw/cours/base-js/dom.html>

Des méthodes pour modifier le DOM:

```
var x = document.getElementById('list')
x.style.color = 'blue';
var items = document.getElementsByTagName('li')
items[0].innerHTML = 'hello'
x.innerHTML += '<li>Nouvel item</li>'
```

...

JavaScript Walkthrough

