Javascript

Javascript

Langage de programmation fonctionnel et orienté objet

Présent dans la majorité des navigateurs web

Permet de manipuler

- Le contenu d'une page web une fois chargée
- Certaines parties du browser
- Valider les champs d'un formulaire

Et ce, *depuis la page elle-même*

Inclusion dans une page web

Par l'intermédiaire de balises `script`

Autant dans le `head` que le `body`

```
<script src="{URL}" />
```

ou

```
<script> {code-javascript} </script>
```

L'attribut `type` est `text/javascript` par défaut

D'autres valeurs peuvent spécifier une version particulière de Javascript

Ou d'autres langages, selon le navigateur

Encodage, encore et toujours

Comment inclure un ‘<’ dans le script?

Bien sûr, même question pour l’élément `style`

Encodage, encore et toujours

Comment inclure un ‘<’ dans le script?

Bien sûr, même question pour l’élément `style`

Ça dépend! En XML ou HTML?

Encodage, encore et toujours

Comment inclure un ‘<’ dans le script?

Bien sûr, même question pour l’élément `style`

Ça dépend! En XML ou HTML?

En HTML, un ‘<’ fonctionne très bien

En XML, il faut utiliser ‘`<`’

¡ Mais ‘`<`’ ne fonctionne pas en HTML !

Le monde merveilleux des hacks

Mettre un ‘<’ dans un script ou style

Qui fonctionne autant avec XML qu’avec HTML

Il nous faut *CDATA* et des commentaires

`<![CDATA[...]]>` (en XML) permet d’inclure du texte avec ‘<’, ‘`&`’

`<![CDATA[<]]>` est équivalent à `<`

Le monde merveilleux des hacks

Mettre un '<' dans un script ou style

Qui fonctionne autant avec XML qu'avec HTML

Il nous faut *CDATA* et des commentaires

`<![CDATA[...]]>` (en XML) permet d'inclure du texte avec '<', '&'

`<![CDATA[<]]>` est équivalent à `<`

```
<script>
  //<![CDATA[
  ...my script..
  //]]>
</script>
```

Calculs en Javascript

```
var x1 = 5 + 6;
var x2 = x1 * 7;
var x3 = x2 - 6 / x1;
```

Abstraire un calcul

```
var pi = 3.141592653589793;
function diameter (radius) {
    return radius * 2;
}
function circumference (radius) {
    return diameter(radius) * pi;
}
function area (x, y) {
    return x * y;
}
```

Chaînes de caractères

```
var name = "BuGit";
var version = 32;
var fullname = name + "-" + String(version);
var fn_length = fullname.length;
```

Encodage, comme toujours

Expressions régulières

```
var re = /b(.*)t/i;
var match_data = fullname.match(re);
var matched_string = match_data[0];
var matched_subgroup = match_data[1];
```

Opérateurs de RegExp

$re*$	Répétition 0 ou plus
$re?$	0 ou 1 fois
$re+$	Répétition 1 ou plus
$re_1 \mid re_2$	Accepte autant l'un que l'autre
[*chars*]	N'importe lequel de ces caractères
[^ *chars*]	N'importe quel autre caractère
(*re*)	Sous-groupe

Note: autre encodage d'une structure arborescente

Objets

```
var bugit_descriptor = {
  name: "BuGit",
  version: 32,
  URL: "http://gitlab.com/monnier/bugit"
};
var bd = bugit_descriptor;
var n = bd.name;
bd.name = "Anonymous";
```

Constructors

```
function SoftDesc (name, version, URL) {
    this.name = name;
    this.version = version;
    this.URL = URL;
}

var bd = new SoftDesc ("BuGit", 32,
        "http://gitlab.com/monnier/bugit")
```

Constructors

```
function SoftDesc (name, version, URL) {
    this.name = name;
    this.version = version;
    this.URL = URL;
}

var bd = new SoftDesc ("BuGit", 32,
        "http://gitlab.com/monnier/bugit")
```

`this`: mot-clé spécial qui contient “l’object courant”

Méthodes

```
function SoftDesc (name, version) {
    this.name = name;
    this.version = version;
    this.fullname = function () {
        return this.name + "-" + this.version;
    }
}

var bd = new SoftDesc ("BuGit", 32);
var bfn = bd.fullname()
```

Délégation

```
function SoftDesc (name, version) {
    this.name = name;
    this.version = version;
}
SoftDesc.prototype.fullname = function () {
    return this.name + "-" + this.version;
}

var bd = new SoftDesc ("BuGit", 32);
var bfn = bd.fullname()
```

Fonctions Javascript

Plus complexes qu'il n'y paraît

Peuvent être utilisées comme

- Fonction
- Constructeur
- Méthode
- Objet

Y compris tout à la fois!