<h1>**Formation web**</h1>

<h2>Partie 3: *JS & TS*[1]</h2>

[1]TS = TypeScript, pas Traitement du Signal enfin...

La syntaxe de base

if, else, for, let, const, ...

learnxinyminutes.com/docs/javascript/

La syntaxe de base

C'est comme Python mais avec plus de symboles inutiles

JavaScript

```
if (condition) {
    actions
} else if (condition) {
    actions
} else {
    actions
}
```

Python

```
if condition:
    actions
elif condition:
    actions
else:
    actions
```

La syntaxe de base

C'est comme Python mais avec plus de symboles inutiles

JavaScript

```
for (const a of truc) {
    actions
}

while (condition) {
    actions
}
```

Python

```
for a in truc:
    actions

while condition:
    actions
```

La syntaxe de base

C’est comme Python mais avec plus de symboles inutiles

JavaScript

```
let a = expression;
const a = expression;

a = expression;
```

Python

```
a = expression
```

n’existe pas

```
a = expression
```

La syntaxe de base

C’est comme Python mais avec plus de symboles inutiles

Python

```
if a == 8:
    actions
```

La syntaxe de base

wtf javascript????

JavaScript

```
if (a === 8) {
    actions
}
```

wtf??????????

wtf????

```
> []==0
< true
```

L’interpolation

`Bienvenue à la forma web n°${formaNum}`

L’interpolation

Construire des chaînes de caractères facilement

`Salut, ${prénom}, tfk?`

L’interpolation

Construire des chaînes de caractères facilement

`**Salut,** ${prénom}**, tfk?**`

L’interpolation

Construire des chaînes de caractères facilement

`Salut, **${prénom}**, tfk?`

L’interpolation

Construire des chaînes de caractères facilement

`Salut, ${prénom}, tfk?`

L’interpolation

Construire des chaînes de caractères facilement

`Salut, ${prénom}, tfk?`

L’interpolation

Construire des chaînes de caractères facilement

```
`Salut, ${4 + 4}, tfk?`
```

Les objets

{ c: “la base”, vraiment: true }

Les objets

Pour stocker des trucs


```
{
  id: 'u:n7xhkqrupqci17kk',
  uid: 'malont',
  createdAt: '2012-09-18T18:17:55.000Z',
  schoolServer: 'inp',
  schoolUid: 'tmalon',
  schoolEmail: 'thierry.malon@toulouse-inp.fr',
  email: 'thierry.malon@toulouse-inp.fr',
  otherEmails: [ [redacted] ],
  firstName: 'Thierry',
  lastName: 'Malon',
  majorId: 'major:cmbikin5prphui4m',
  minorId: null,
  graduationYear: 2015,
  apprentice: false,
  address: [redacted],
  birthday: '1993-04-19T00:00:00.000Z',
  description: '',
  nickname: '',
  phone: [redacted],
  pictureFile: 'users/malont.png',
  godparentId: 'u:jjuwlaieneejlp0t',
  cededImageRightsToTVn7: true,
  enabledNotificationChannels: [ 'Articles', 'Shotguns', 'Permissions',
'GroupBoard', 'GodparentRequests', 'Comments', 'Other' ],
  admin: false,
  canEditUsers: false,
  canEditGroups: false,
  canAccessDocuments: true
}
```

Le JSON

C’est du JavaScript!


```
{
  "id": "u:n7xhkqrupqcil7kk",
  "uid": "malont",
  "createdAt": "2012-09-18T18:17:55.000Z",
  "schoolServer": "inp",
  "schoolUid": "tmalon",
  "schoolEmail": "thierry.malon@toulouse-inp.fr",
  "email": "thierry.malon@toulouse-inp.fr",
  "otherEmails": [redacted]" ],
  "firstName": "Thierry",
  "lastName": "Malon",
  "majorId": "major:cmbikin5prphui4m",
  "minorId": null,
  "graduationYear": 2015,
  "apprentice": false,
  "address": [redacted],
  "birthday": "1993-04-19T00:00:00.000Z",
  "description": "",
  "nickname": "",
  "phone": [redacted],
  "pictureFile": "users/malont.png",
  "godparentId": "u:jjuwlaieneejlp0t",
  "cededImageRightsToTVn7": true,
  "enabledNotificationChannels": [ "Articles", "Shotguns", "Permissions",
"GroupBoard", "GodparentRequests", "Comments", "Other" ],
  "admin": false,
  "canEditUsers": false,
  "canEditGroups": false,
  "canAccessDocuments": true
}
```

Le JSON

C’est du JavaScript!

```
let meilleurProf = {
  "id": "u:n7xhkqrupqcil7kk",
  "uid": "malont",
  "createdAt": "2012-09-18T18:17:55.000Z",
  "schoolServer": "inp",
  "schoolUid": "tmalon",
  "schoolEmail": "thierry.malon@toulouse-inp.fr",
  "email": "thierry.malon@toulouse-inp.fr",
  "otherEmails": [ ],
  "firstName": "Thierry",
  "lastName": "Malon",
  "majorId": "major:cmbikin5prphui4m",
  "minorId": null,
  "graduationYear": 2015,
  "apprentice": false,
  "address": "
  "birthday": "1993-04-19T00:00:00.000Z",
  "description": "",
  "nickname": "",
  "phone":  ,
  "pictureFile": "users/malont.png",
  "godparentId": "u:jjuwlaieneejlp0t",
  "cededImageRightsToTVn7": true,
  "enabledNotificationChannels": [ "Articles", "Shotguns", "Permissions",
"GroupBoard", "GodparentRequests", "Comments", "Other" ],
  "admin": false,
  "canEditUsers": false,
  "canEditGroups": false,
  "canAccessDocuments": true
}
```

Les objets

Pour stocker des trucs

```
const meilleurProf = { ... }
```

Les objets

Pour stocker des trucs

```
const meilleurProf = { ... }
const enseeiht =  {  }
```

Les objets

Pour stocker des trucs

```
const meilleurProf = { ... }
const enseeiht = {
   profs: [ ... ],
}
```

Les objets

Pour stocker des trucs

```
const meilleurProf = { ... }
const enseeiht = {
   profs: [ ... ],
   meilleurProf:
}
```

Les objets

Pour stocker des trucs

```
const meilleurProf = { ... }
const enseeiht = {
   profs: [ ... ],
   meilleurProf: meilleurProf
}
```

Les objets

Pour stocker des trucs

```
const meilleurProf = { ... }
const enseeiht = {
   profs: [ ... ],
   meilleurProf: meilleurProf
}
```

Les objets

Pour stocker des trucs

```
const meilleurProf = { ... }
const enseeiht = {
   profs: [ ... ],
   meilleurProf
}
```

Les objets

Pour stocker des trucs

```
const meilleurProf = { ... }
const enseeiht = {
   profs: [ ... ],
   meilleurProf: meilleurProf
}
```

La déstructuration

~~De la société~~

```
const data = n7data()
const meilleurProf = data.meilleurProf
```

La déstructuration

~~De la société~~

```
const data = n7data()
// { profs: [ ... ], meilleurProf: { ... } }
const meilleurProf = data.meilleurProf
```

La déstructuration

~~De la société~~

```
const { meilleurProf } = n7data()
// { profs: [ ... ], meilleurProf: { ... } }
const meilleurProf = data.meilleurProf
```

La déstructuration

~~De la société~~

```
const { meilleurProf } = n7data()
```

La déstructuration

~~De la société~~

```
const téléphone = “+33 01 23 45 67 89”
```

La déstructuration

~~De la société~~

```
const téléphone = “+33 01 23 45 67 89”
téléphone.split(“ ”)
// [“+33”, “01”, “23”, “45”, “67”, “89”]
```

La déstructuration

~~De la société~~

```
const téléphone = “+33 01 23 45 67 89”
const parties = téléphone.split(“ ”)
```

La déstructuration

~~De la société~~

```
const téléphone = “+33 01 23 45 67 89”
const parties = téléphone.split(“ ”)
const préfixe = parties[0]
```

La déstructuration

~~De la société~~

```
const téléphone = “+33 01 23 45 67 89”
const parties = téléphone.split(“ ”)
const préfixe = parties[0]
const reste = [
   parties[1],
]
```

La déstructuration

~~De la société~~

```
const téléphone = “+33 01 23 45 67 89”
const parties = téléphone.split(“ ”)
const préfixe = parties[0]
const reste = [
    parties[1],
    parties[2],
]
```

La déstructuration

~~De la société~~

```
const téléphone = “+33 01 23 45 67 89”
const parties = téléphone.split(“ ”)
const préfixe = parties[0]
const reste = [
    parties[1],
    parties[2],
    parties[3],
    parties[4],
    parties[5]
]
```

La déstructuration

~~De la société~~

```
const téléphone = "+33 01 23 45 67 89"
const parties = téléphone.split(" ")
const préfixe = parties[0]
const reste = [
    parties[1],
    parties[2],
    parties[3],
    parties[4],
    parties[5]
]
```

La déstructuration

~~De la société~~

```
const téléphone = "+33 01 23 45 67 89"
const [ préfixe,  ] = téléphone.split(" ")
      ["+33", "01", "23", "45", "67", "89"]
const reste = [
    parties[1],
    parties[2],
    parties[3],
    parties[4],
    parties[5]
]
```

La déstructuration

~~De la société~~

```
const téléphone = “+33 01 23 45 67 89”
const [ préfixe, ...reste ] = téléphone.split(“ ”)
      [“+33”, “01”, “23”, “45”, “67”, “89”]
```

La déstructuration

~~De la société~~

```
const téléphone = “+33 01 23 45 67 89”
const [ préfixe, ...reste ] = téléphone.split(“ ”)
```

?.

Optional chaining

?.

Optional chaining

```
{
  id: 'u:n7xhkqrupqcil7kk',
  uid: 'malont',
  email: 'thierry.malon@toulouse-inp.fr',
  firstName: 'Thierry',
  lastName: 'Malon',
  major: { name: ‘SN’ },
  graduationYear: 2015,
  apprentice: false,
  birthday: '1993-04-19T00:00:00.000Z',
}
```

?.

Optional chaining

```
',

oulouse-inp.fr',

                let { firstName, major } = eleve
                print(`${firstName} est à l'${major.name.school}`)

0:00:00.000Z',
```

?.

Optional chaining

```
{
  id: 'u:n7xhkqrupqcil7kk',
  uid: 'malont',
  email: 'thierry.malon@toulouse-inp.fr',
  firstName: 'Thierry',
  lastName: 'Malon',
  major: { name: 'SN', school: { name: 'ENSEEIHT' } },
  graduationYear: 2015,
  apprentice: false,
  birthday: '1993-04-19T00:00:00.000Z',
}
```

?.

Optional chaining

```
{
  id: 'u:n7xhkqrupqcil7kk',
  uid: 'malont',
  email: 'thierry.malon@toulouse-inp.fr',
  firstName: 'Thierry',
  lastName: 'Malon',
  major: null,
  graduationYear: 2015,
  apprentice: false,
  birthday: '1993-04-19T00:00:00.000Z',
}
```

?.

Optional chaining

```
',

oulouse-inp.fr',

                let { firstName, major } = eleve
                print(`${firstName} est à l' ${null.school.name}`)

0:00:00.000Z',
```

?.

Optional chaining

```
',
oulouse-inp.fr',
0:00:00.000Z',
```

```
let { firstName, major } = eleve
print(`${firstName} est à l' ${null.school.name}`)
```

Uncaught TypeError: Cannot read properties of null (reading 'school')

?.

Optional chaining

```
',
oulouse-inp.fr',
                let { firstName, major } = eleve
                print(`${firstName} est à l' ${null.school.name}`)
0:00:00.000Z',
```

Uncaught TypeError: Cannot read properties of null (reading 'school')

?.

Optional chaining

```
',

oulouse-inp.fr',

                let { firstName, major } = eleve
                print(`${firstName} est à l' ${major.school.name}`)

0:00:00.000Z',
```

?.

Optional chaining

```
',

oulouse-inp.fr',

                let { firstName, major } = eleve
                print(`${firstName} est à l' ${major?.school.name}`)

0:00:00.000Z',
```

?.

Optional chaining

```
let { firstName, major } = eleve
print(`${firstName} est à l' ${{...}?.school.name}`)
```

“n7”

?.

Optional chaining

```
let { firstName, major } = eleve
print(`${firstName} est à l' ${null?.school.name}`)
```

undefined

?.

Optional chaining

Ewen est à l’n7

?.

Optional chaining

Lionel est à l'**undefined**

Guilhem: Malaisant, je fais une issue

??

Nullish coalescing operator

??

Nullish coalescing operator

truc qui peut être undefined ou null ?? valeur par défaut

??

Nullish coalescing operator

```
major?.school.name ?? 'a retraite'
```

??

Nullish coalescing operator

```
major?.school.name ?? 'a retraite'
```

Lionel est à l'a retraite

Un ptit résumé Des trucs chelous en JavaScript

`{ truc }`	`{ truc: truc }`
`let { a, ...reste } = b`	`let a = b.a` `let reste = b mais sans a`

`let [ a, b, ...c ] = d`	`let a = d[0]`
	`let b = d[1]`
	`let c = d mais sans les 2 premiers éléments`

`a = b?.c.d`	`if (b) { a = b.c.d } else { a = undefined }`
`a = b ?? c`	`if (b) { a = b } else { a = c }`
`a === b`	`a == b, mais fais pas de conversions cheloues stp`
`` `salut, ${a}.` ``	`“salut, “ + a.toString() + “.”`

Un ptit résumé Des trucs chelous en JavaScript

`{ truc }`	`{ truc: truc }`
`let { a, ...reste } = b`	`let a = b.a` `let reste = b mais sans a`

`let [ a, b, ...c ] = d`	`let a = d[0]`
	`let b = d[1]`
	`let c = d mais sans les 2 premiers éléments`

`a = b?.c.d`	`if (b) { a = b.c.d } else { a = undefined }`
`a = b ?? c`	`if (b) { a = b } else { a = c }`
`a === b`	`a == b, mais fais pas de conversions cheloues stp`
`` `salut, ${a}.` ``	`“salut, “ + a.toString() + “.”`

Un ptit résumé Des trucs chelous en JavaScript

`{ truc }`	`{ truc: truc }`
`let { a, ...reste } = b`	`let a = b.a` `let reste = b mais sans a`
`let [ a, b, ...c ] = d`	**`let a = d[0]`** **`let b = d[1]`** **`let c = d mais sans les 2 premiers éléments`**
`a = b?.c.d`	`if (b) { a = b.c.d } else { a = undefined }`
`a = b ?? c`	`if (b) { a = b } else { a = c }`
`a === b`	`a == b, mais fais pas de conversions cheloues stp`
`` `salut, ${a}.` ``	`“salut, “ + a.toString() + “.”`

Un ptit résumé Des trucs chelous en JavaScript

`{ truc }`	`{ truc: truc }`
`let { a, ...reste } = b`	`let a = b.a` `let reste = b mais sans a`

`let [ a, b, ...c ] = d`	`let a = d[0]`
	`let b = d[1]`
	`let c = d mais sans les 2 premiers éléments`

`a = b?.c.d`	`if (b) { a = b.c.d } else { a = undefined }`
`a = b ?? c`	`if (b) { a = b } else { a = c }`
`a === b`	`a == b, mais fais pas de conversions cheloues stp`
`` `salut, ${a}.` ``	`“salut, “ + a.toString() + “.”`

Un ptit résumé Des trucs chelous en JavaScript

`{ truc }`	`{ truc: truc }`
`let { a, ...reste } = b`	`let a = b.a` `let reste =` *`b mais sans a`*

`let [ a, b, ...c ] = d`	`let a = d[0]`
	`let b = d[1]`
	`let c =` *`d mais sans les 2 premiers éléments`*

`a = b?.c.d`	`if (b) { a = b.c.d } else { a = undefined }`
`a = b ?? c`	**`if (b) { a = b } else { a = c }`**
`a === b`	`a == b,` *`mais fais pas de conversions cheloues stp`*
`` `salut, ${a}.` ``	`“salut, “ + a.toString() + “.”`

Un ptit résumé Des trucs chelous en JavaScript

`{ truc }`	`{ truc: truc }`
`let { a, ...reste } = b`	`let a = b.a` `let reste =` *`b mais sans a`*

`let [ a, b, ...c ] = d`	`let a = d[0]`
	`let b = d[1]`
	`let c =` *`d mais sans les 2 premiers éléments`*

`a = b?.c.d`	`if (b) { a = b.c.d } else { a = undefined }`
`a = b ?? c`	`if (b) { a = b } else { a = c }`
`a === b`	**`a == b`, *`mais fais pas de conversions cheloues stp`***
`` `salut, ${a}.` ``	`“salut, “ + a.toString() + “.”`

Un ptit résumé Des trucs chelous en JavaScript

`{ truc }`	`{ truc: truc }`
`let { a, ...reste } = b`	`let a = b.a` `let reste =` *`b mais sans a`*

`let [ a, b, ...c ] = d`	`let a = d[0]`
	`let b = d[1]`
	`let c =` *`d mais sans les 2 premiers éléments`*

`a = b?.c.d`	`if (b) { a = b.c.d } else { a = undefined }`
`a = b ?? c`	`if (b) { a = b } else { a = c }`
`a === b`	`a == b,` *`mais fais pas de conversions cheloues stp`*
`` `salut, ${a}.` ``	**`“salut, “ + a.toString() + “.”`**

Un ptit résumé Des trucs chelous en JavaScript

```
{ truc }
```

```
{ truc: truc }
```

```
let { a, ...reste } = b
```

```
let a = b.a
let reste = b mais sans a
```

```
let [ a, b, ...c ] = d
```

```
let a = d[0]
let b = d[1]
let c = d mais sans les 2 premiers éléments
```

```
a = b?.c.d
```

```
if (b) { a = b.c.d } else { a = undefined }
```

```
a = b ?? c
```

```
if (b) { a = b } else { a = c }
```

```
a === b
```

```
a == b, mais fais pas de conversions cheloues stp
```

```
`salut, ${b}.`
```

```
“salut, “ + b.toString() + “.”
```

Le DOM

Manipuler son HTML

Le DOM

Manipuler son HTML

```
<balise attribut=”valeur”>contenu ...</balise>
```

Le DOM

Manipuler son HTML

```
<balise attribut="valeur">contenu ...</balise>
```

Le DOM

Manipuler son HTML

```
<balise attribut="valeur">contenu...</balise>
```

Le DOM

Manipuler son HTML

```
<balise attribut="valeur">contenu...</balise>
```

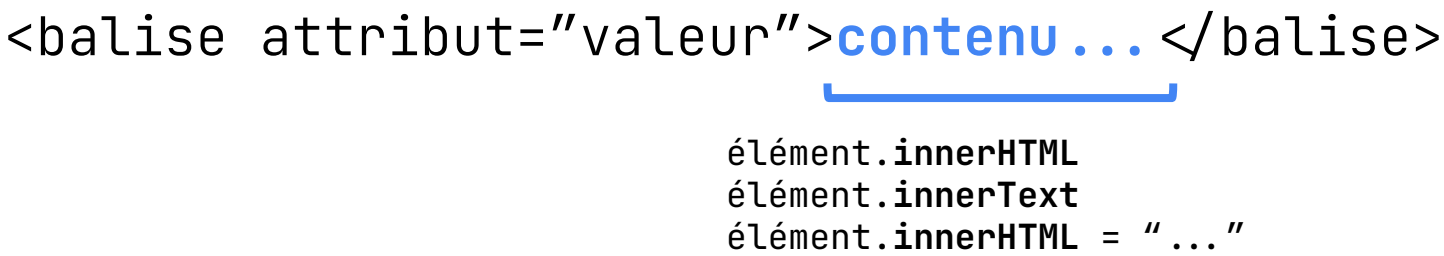
Le DOM

Manipuler son HTML


```
élément.getAttribute("attribut")
élément.setAttribute("attribut", "nouvelle valeur")
élément.removeAttribute("attribut")
```

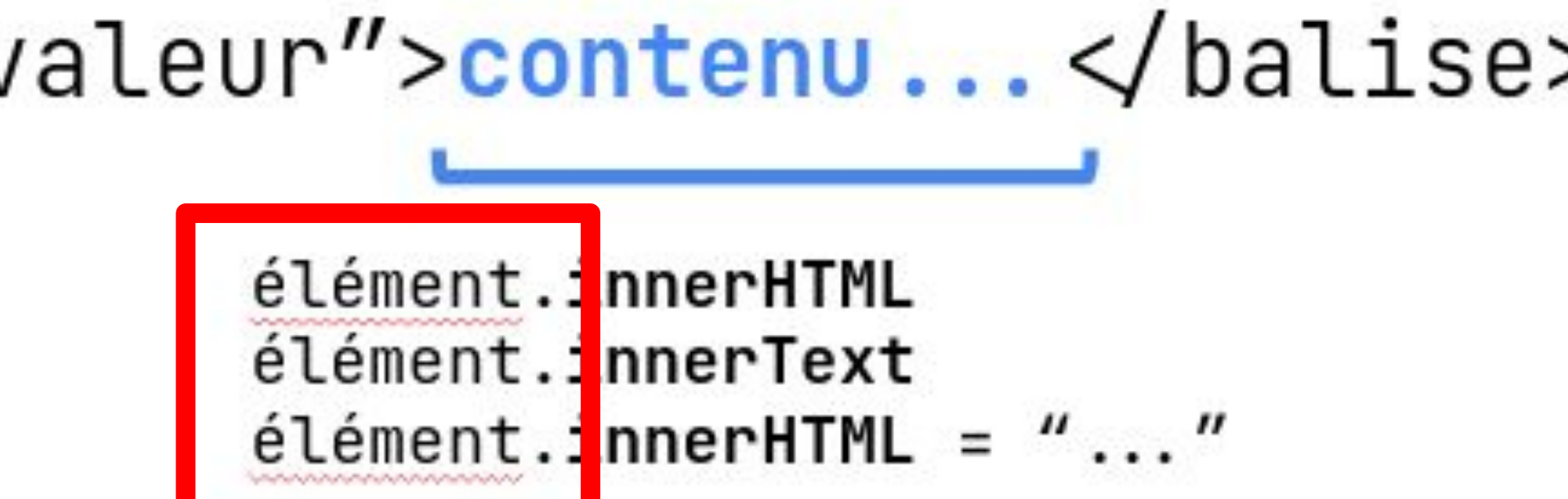
Le DOM

Manipuler son HTML



valeur”>contenu ...</balise>

élément.**innerHTML**
élément.**innerText**
élément.**innerHTML** = “...”


valeur”>contenu ...</balise>
élément.innerHTML
élément.innerText
élément.innerHTML = “...”
???

Le DOM

Manipuler son HTML

```
<balise attribut=”valeur”>
   contenu...
</balise>
```

Le DOM

Manipuler son HTML

```
<balise id="unTrucUnique" attribut="valeur">
   contenu...
</balise>
```

Le DOM

Manipuler son HTML

```
<balise id="unTrucUnique" attribut="valeur">
   contenu...
</balise>
```

```
const élément = document.getElementById("unTrucUnique")
```

Le DOM

Manipuler son HTML

```
<h1 class="title" attribut="valeur">
    contenu...
</h1>
```

```
const élément = document.querySelector("h1.title")
```

Le DOM

Manipuler son HTML

```
<h1 class="title" attribut="valeur">
    contenu...
</h1>

<h1 class="title secondary" attribut="valeur">
    contenu...
</h1>
```

```
const éléments = document.querySelectorAll("h1.title")
```

Le DOM

Manipuler son HTML

```
aleur”>
```

```
ttribut=”valeur”>
```

```
electorAll(“h1.title”)
```

```
[
    <h1 class=”title”>...</h1>,
    <h1 class=”title secondary”>...</h1>
]
```

Le DOM

Manipuler son HTML

```
const éléments = [
    <h1 class="title">...</h1>,
    <h1 class="title secondary">...</h1>
]
```

(si mais c'est du JSX et pas le temps)

`.addEventListener`

Réagir à des trucs

.addEventListener

Réagir à des trucs

```
<button>Sommaire</button>

<ul class=”hidden”>
   <li>Partie I</li>
   <li>Partie II</li>
   <li>Partie III</li>
</ul>
```

.addEventListener

Réagir à des trucs

```
<button id="toggle-toc">Sommaire</button>

<ul class="hidden" id="toc">
    <li>Partie I</li>
    <li>Partie II</li>
    <li>Partie III</li>
</ul>
```

.addEventListener

Réagir à des trucs

```
<button id="toc-toggle">Sommaire</button>

<ul class="hidden" id="toc">
    <li>Partie I</li>
    <li>Partie II</li>
    <li>Partie III</li>
</ul>


const tocToggle = document.getElementById("toc-toggle")
```

.addEventListener

Réagir à des trucs

```
<button id=”toc-toggle”>Sommaire</button>

<ul class=”hidden” id=”toc”>
    <li>Partie I</li>
    <li>Partie II</li>
    <li>Partie III</li>
</ul>


const tocToggle = document.getElementById(“toc-toggle”)
const toc = document.getElementById(“toc”)
```

.addEventListener

Réagir à des trucs

```
const tocToggle = document.getElementById(“toc-toggle”)
const toc = document.getElementById(“toc”)



tocToggle.addEventListener(“click”, () => {
    toc.setAttribute(“class”, “hidden”)
})
```

.addEventListener

Réagir à des trucs

```
const tocToggle = document.getElementById("toc-toggle")
const toc = document.getElementById("toc")


tocToggle.addEventListener("click", () => {
    toc.setAttribute("class", "hidden")
})
```

Les callbacks

Ouais, j’te *rappelle* quand j’ai fini avec ça

Les callbacks

Ouais, j’te rappelle quand j’ai fini avec ça

```
truc.addEventListener(“évènement”, action)
```

Les callbacks

Ouais, j'te rappelle quand j'ai fini avec ça

```
truc.addEventListener("évènement", action)
```

une fonction !

Les callbacks

Ouais, j’te rappelle quand j’ai fini avec ça

```
function action() {
   console.log(“bonjour :)”)
}

truc.addEventListener(“évènement”, action)
```

Les callbacks

Ouais, j’te rappelle quand j’ai fini avec ça

```
function action() {
   console.log(“bonjour :)”)
}

truc.addEventListener(“évènement”, () ⇒ {
   console.log(“bonjour :)”)
})
```

Les callbacks

Ouais, j'te rappelle quand j'ai fini avec ça

```
function action() {
   console.log("bonjour :)")
}

truc.addEventListener("évènement", () => {
   console.log("bonjour :)")
})
```

De la pratique

Eh bah putain, c'est pas trop tôt

→ net7.dev/formaweb-3-dom

https://codepen.io/pen/oNmBoKg

Typescript

Préciser explicitement les types de nos variables

```
failbowl:~(master!?) $ jsc
> [] + []

> [] + {}
[object Object]
> {} + []
0
> {} + {}
NaN
>
```

```
failbowl:~(master!?) $ jsc
> [] + []
> [] + {}
[object Object]
> {} + []
0
> {} + {}
NaN
>
```

liste + liste = chaîne de caractères

```
failbowl:~(master!?) $ jsc
> [] + []
> [] + {}
[object Object]
> {} + []
0
> {} + {}
NaN
>
```

liste + liste = chaîne de caractères

liste + objet = objet

```
failbowl:~(master!?) $ jsc
> [] + []
> [] + {}
[object Object]
> {} + []
0
> {} + {}
NaN
>
```

liste + liste = chaîne de caractères

liste + objet = objet

objet + liste = zéro ???

```
failbowl:~(master!?) $ jsc
> [] + []
> [] + {}
[object Object]
> {} + []
0
> {} + {}
NaN
>
```

liste + liste = chaîne de caractères

liste + objet = objet

objet + liste = zéro ???

objet + objet = NaN (Not a Number) ???

Console | What's New

top | Filter

```
> 2 + 2
4
> "2" + "2"
"22"
> 2 + 2 - 2
2
> "2" + "2" - "2"
20
```

Typescript

Ajouter des types

```
function decodeApogeeCode(code) {
	const pattern = /N([56789])(EE|AE|EK|AN|EN|EAN|EM)(\d+)([A-Z])?/;
	const match = pattern.exec(code);
	if (!match) {
		return undefined;
	}
	const [_, semester, major, unitNumber, subjectLetter] = match;
	if (!semester || !major || !unitNumber) {
		return undefined;
	}

	return {
		semester: parseInt(semester), major, subjectLetter,
		unitNumber: parseInt(number)
	}
}
```

Typescript

Ajouter des types

```
function decodeApogeeCode(code: string): ApogeeCode | undefined {
     const pattern = /N([56789])(EE|AE|EK|AN|EN|EAN|EM)(\d+)([A-Z])?/;
     const match = pattern.exec(code);
     if (!match) {
          return undefined;
     }
     const [_, semester, major, unitNumber, subjectLetter] = match;
     if (!semester || !major || !unitNumber) {
          return undefined;
     }

     return {
          semester: parseInt(semester), major, subjectLetter,
          unitNumber: parseInt(number)
     }
}

type ApogeeCode = {
     semester: number; major: string; unitNumber: number; subjectLetter: string;
}
```

Typescript

Ajouter des types

```
function decodeApogeeCode(code: string): ApogeeCode | undefined {
    const pattern = /N([56789])(EE|AE|EK|AN|EN|EAN|EM)(\d+)([A-Z])?/;
    const match = pattern.exec(code);
    if (!match) {
        return undefined;
    }
    const [_, semester, major, unitNumber, subjectLetter] = match;
    if (!semester || !major || !unitNumber) {
        return undefined;
    }

    return {
        semester: parseInt(semester), major, subjectLetter,
        unitNumber: parseInt(number)
    }
}

type ApogeeCode = {
    semester: number; major: string; unitNumber: number; subjectLetter: string;
}
```

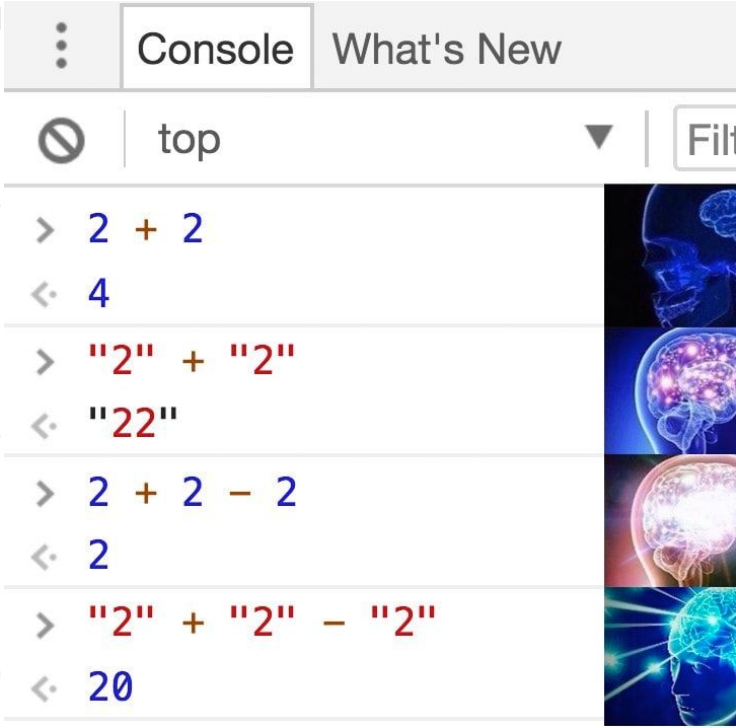
Typescript

Ajouter des types

```
function decodeApogeeCode(code: string): ApogeeCode | undefined {
    const pattern = /N([56789])(EE|AE|EK|AN|EN|EAN|EM
    const match = pattern.exec(code);
    if (!match) {
        return undefined;
    }
    const [_, semester, major, unitNumber, subjectLet
    if (!semester || !major || !unitNumber) {
        return undefined;
    }

    return {
        semester: parseInt(semester), major, subject
        unitNumber: parseInt(number)
    }
}

type ApogeeCode = {
    semester: number; major: string; unitNumber: numb
}
```



Typescript

Ajouter des types

```
function decodeApogeeCode(code: string): ApogeeCode | undefined {
    const pattern = /N([56789])(EE|AE|EK|AN|EN|EAN|EM)(\d+)([A-Z])?/;
    const match = pattern.exec(code.|
```

```
⬡ toLowerCase
⬡ replace
⬡ split
⬡ indexOf
□ length
⌾ Symbol
⬡ at
⬡ charAt
⬡ charCodeAt
⬡ codePointAt
⬡ concat
```

```
type ApogeeCode = {
    semester: number; major: string; unitNumber: number; subjectLetter: string;
}
```

Petit inventaire de types

Les plus utiles

Petit inventaire de types

Les plus utiles

```
string, number, bool, null,
undefined
```

```
“gjroger”, 89283, true, null, undefined
```

Petit inventaire de types

Les plus utiles

`string, number, bool, null, undefined`	`“gjroger”, 89283, true, null, undefined`
`string[] ou Array<string>`	`[ “coubeh”, “feur” ]`

Petit inventaire de types

Les plus utiles

`string, number, bool, null, undefined`	`“gjroger”, 89283, true, null, undefined`
`string[] ou Array<string>`	`[ “coubeh”, “feur” ]`
`Record<string, number>`	`{ ❤️: 4, 😂: 67 }`

Petit inventaire de types

Les plus utiles

`string, number, bool, null, undefined`	`“gjroger”, 89283, true, null, undefined`
`string[] ou Array<string>`	`[ “coubeh”, “feur” ]`
`Record<string, number>`	`{ ❤️: 4, 😂: 67 }`
`{ quoi: string; }`	`{ quoi: “feur” }`

Petit inventaire de types

Les plus utiles

`string, number, bool, null, undefined`	`“gjroger”, 89283, true, null, undefined`
`string[] ou Array<string>`	`[ “coubeh”, “feur” ]`
`Record<string, number>`	`{ ❤️: 4, 😂: 67 }`
`{ quoi: string; }`	`{ quoi: “feur” }`

`(user: User, event: Event) ⇒ bool`	`(u, e) ⇒ e.managers.some(`
	`  m ⇒ m.uid ═══ u.uid)`
	`)`

Petit inventaire de types

Les plus utiles

`string, number, bool, null, undefined`	`“gjroger”, 89283, true, null, undefined`
`string[] ou Array<string>`	`[ “coubeh”, “feur” ]`
`Record<string, number>`	`{ ❤️: 4, 😂: 67 }`
`{ quoi: string; }`	`{ quoi: “feur” }`

`(user: User, event: Event) ⇒ bool`	`(u, e) ⇒ e.managers.some(`
	`  m ⇒ m.uid ═══ u.uid)`
	`)`

`{ quoi: string } & { hein: string }`	`{ quoi: “feur”, hein: “apagnan” }`

Petit inventaire de types

Les plus utiles

`string, number, bool, null, undefined`	`“gjroger”, 89283, true, null, undefined`
`string[] ou Array<string>`	`[ “coubeh”, “feur” ]`
`Record<string, number>`	`{ ❤️: 4, 😂: 67 }`
`{ quoi: string; }`	`{ quoi: “feur” }`

`(user: User, event: Event) ⇒ bool`	`(u, e) ⇒ e.managers.some(`
	`  m ⇒ m.uid ⩵ u.uid)`
	`)`

`{ quoi: string } & { hein: string }`	`{ quoi: “feur”, hein: “apagnan” }`
`{ quoi: string } | { hein: string }`	`{ quoi: “feur”} ou { hein: “apagnan” }`

Les génériques

Le medicament pas cher

Les génériques

Le medicament pas cher

```
function sort(array) {
   const out = [...array]
   out.sort()
   return out
}
```

Les génériques

Le medicament pas cher

```
function sort(array: number[]): number[] {
   const out = [...array]
   out.sort()
   return out
}
```

Les génériques

Le medicament pas cher

```
function sort(array: number[]): number[] {
    const out = [...array]
    out.sort()
    return out
}
```

```
sort(["a", "b", "c"])
```

```
TypeScript Error: Mega cringe
```

Les génériques

Le medicament pas cher

```
function sort(array: ce que tu veux[]): la même[] {
   const out = [...array]
   out.sort()
   return out
}
```

Les génériques

Le medicament pas cher

```
function sort(array: V[]): V[] {
   const out = [...array]
   out.sort()
   return out
}
```

Les génériques

Le medicament pas cher

```
function sort<V>(array: V[]): V[] {
   const out = [...array]
   out.sort()
   return out
}
```

Les génériques

Le medicament pas cher

```
type Array<T> = T[]
```

Les enums

Pour quand on a plusieurs valeurs particulières

Les enums

Un ensemble de différentes valeurs possibles

```
type Registration = {
   ticket: Ticket,
   state:  payée ou pas payée ou annulée ou opposée
}
```

Les enums

Un ensemble de différentes valeurs possibles

```
type Registration = {
   ticket: Ticket,
   state: “paid” | “unpaid” | “cancelled” |
“opposed”
}
```

Les enums

Un ensemble de différentes valeurs possibles

```
type Registration = {
   ticket: Ticket,
   state: RegistrationState
}

enum RegistrationState {
   Unpaid, Paid, Cancelled, Opposed
}
```

Des ressources

developer.mozilla.org

typescriptlang.org

Merci!

À la prochaine pour faire du Svelte