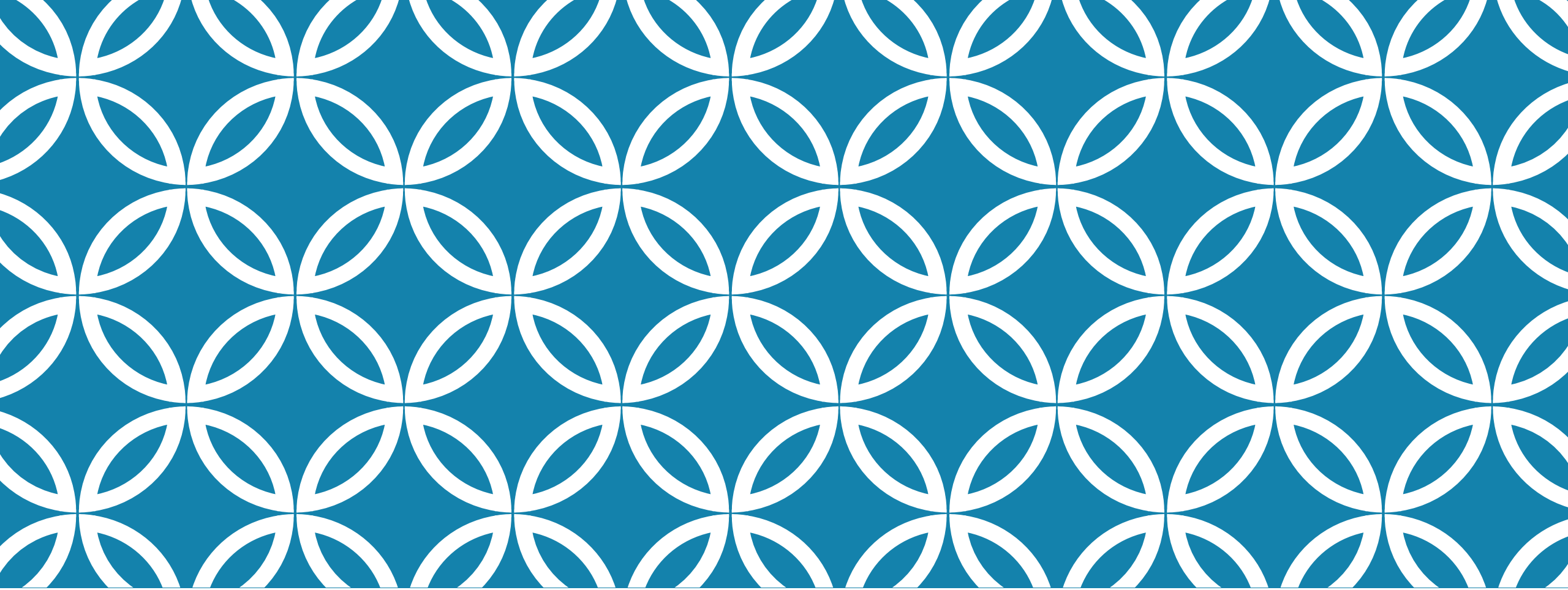
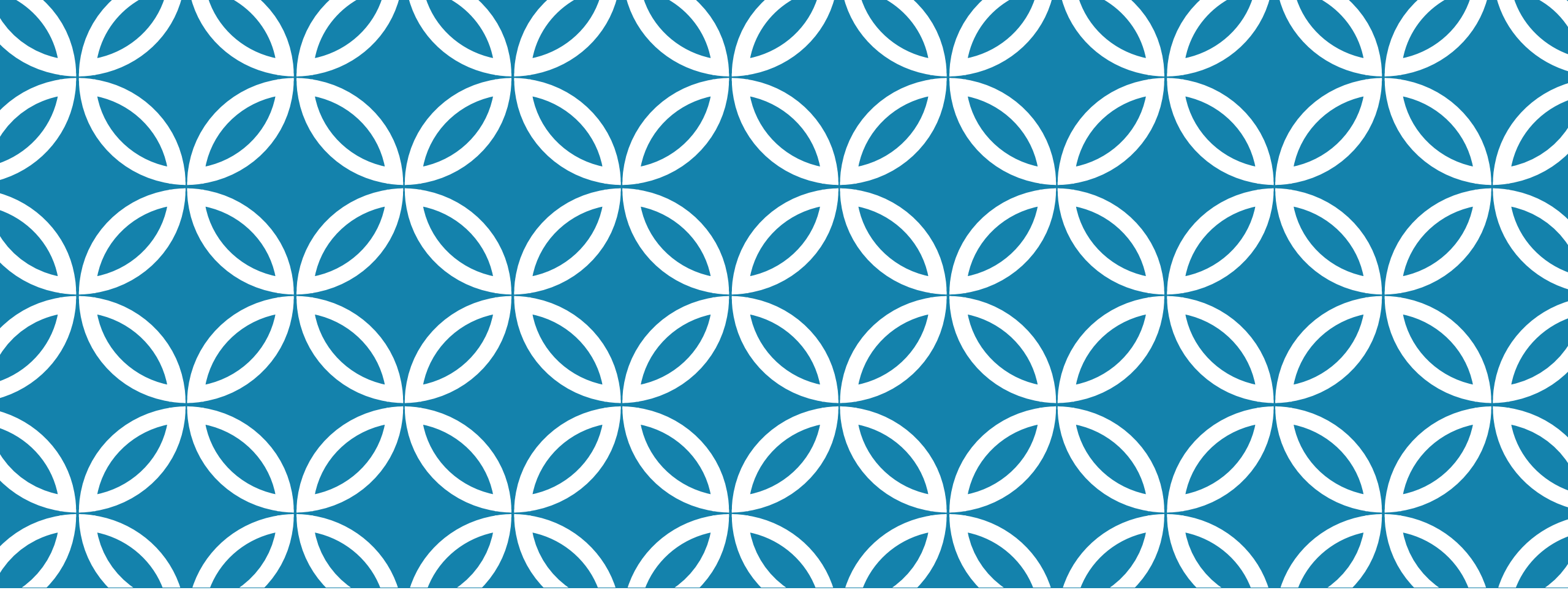


FORMATION JAVASCRIPT

Jordan ABID



INTRODUCTION

LES COMPOSANTS DE JAVASCRIPT

JavaScript est un langage de programmation de scripts principalement utilisé dans les pages web interactives (comme par exemple, la validation de données d'un formulaire). Il est formé de trois composants :

- **ECMAScript,** qui est défini dans l'édition ECMA-262, et qui fournit les fonctionnalités centrales
- **DOM (Document Object Model)** qui fournit les fonctionnalités pour interagir avec le contenu d'une page web
- **BOM (Browser Object Model)** qui fournit les fonctionnalités pour interagir avec le navigateur

JAVASCRIPT DANS HTML

L'élément HTML script permet d'intégrer du code JavaScript dans une page.

Les attributs de cet élément sont :

- **type** : indique le type de contenu (appelé aussi type MIME).
  La valeur est typiquement "text/Javascript".
- **charset** (optionnel) : indique le jeu de caractères utilisé.
- **defer** (optionnel) : indique si l'exécution du script doit être décalée.
- **src** (optionnel) : indique que le code se situe dans un fichier externe

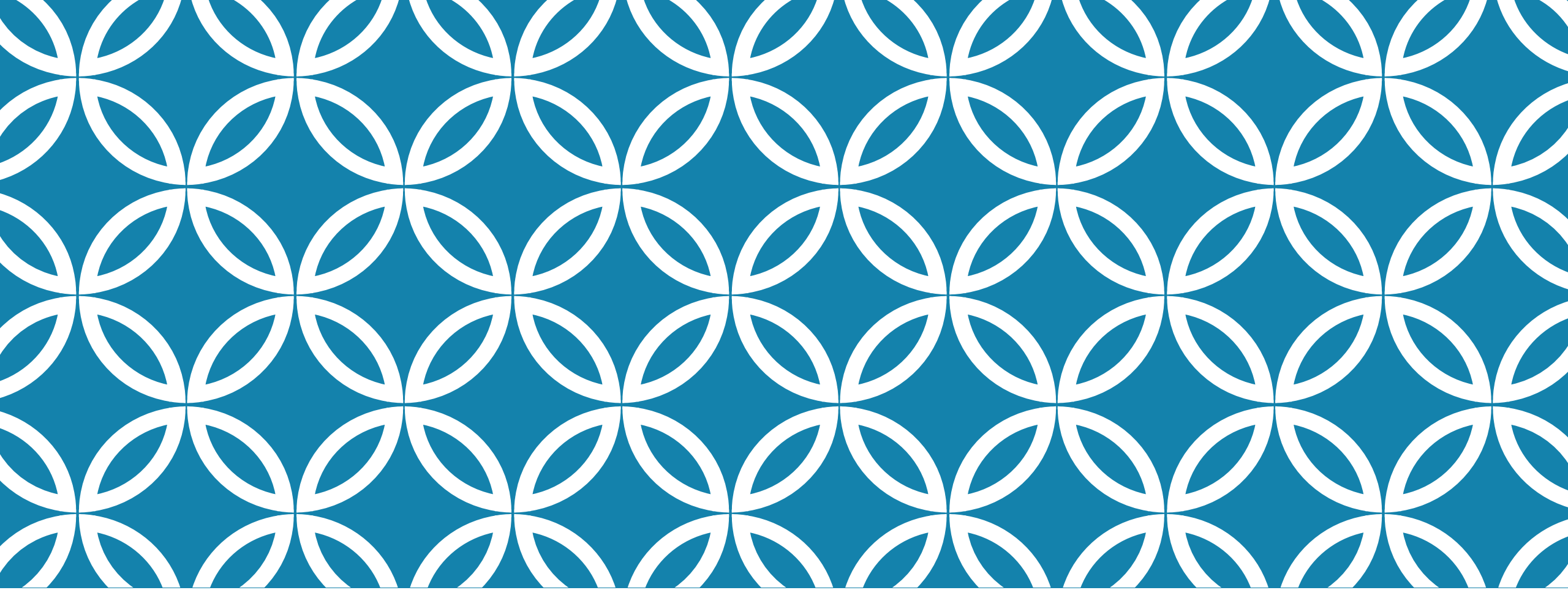
INLINE CODE / EXTERNAL FILE

Inline : Il suffit d'utiliser uniquement l'attribut type et de placer le code au cœur de l'élément script.

```
<script type="text/javascript">
      function sayHi() {alert("Hi!"); }
</script>
```

External : Il suffit d'utiliser uniquement l'attribut type avec l'attribut src.

```
<script type="text/javascript" src="example2.js"/>
```



BASES DU LANGAGE

LES IDENTIFICATEURS ET COMMENTAIRES

Les identificateurs

– sont sensibles a la casse: **test** n'est pas **Test**

– sont formés par convention en utilisant le style camel case comme dans *sommeNotes*

Les instructions
– se terminent par un point-virgule
– nécessitent des accolades lorsqu'elles forment un bloc

```
// single line comment
/* multi-line comment */
```

```
var sum = a + b;
if (sum > 0) alert("positif");
```

FONCTIONS SUR LES STRING

Il existe de nombreuses fonctions sur les chaînes de caractères.

```
var s = "hello world";
alert(s.length);                // 11
alert(s.charAt(1));             // "e"
alert(s.charCodeAt(1));         // 101
alert(s.slice(3));              // "lo world"
alert(s.slice(-3));             // "rld"
alert(s.substring(3,7));        // "lo w"
alert(s.indexOf("o"));          // 4
alert(s.lastIndexOf("o"));      // 7
alert(s.toUpperCase());         // HELLO WORLD
alert(s + " !");                // hellow world !
```

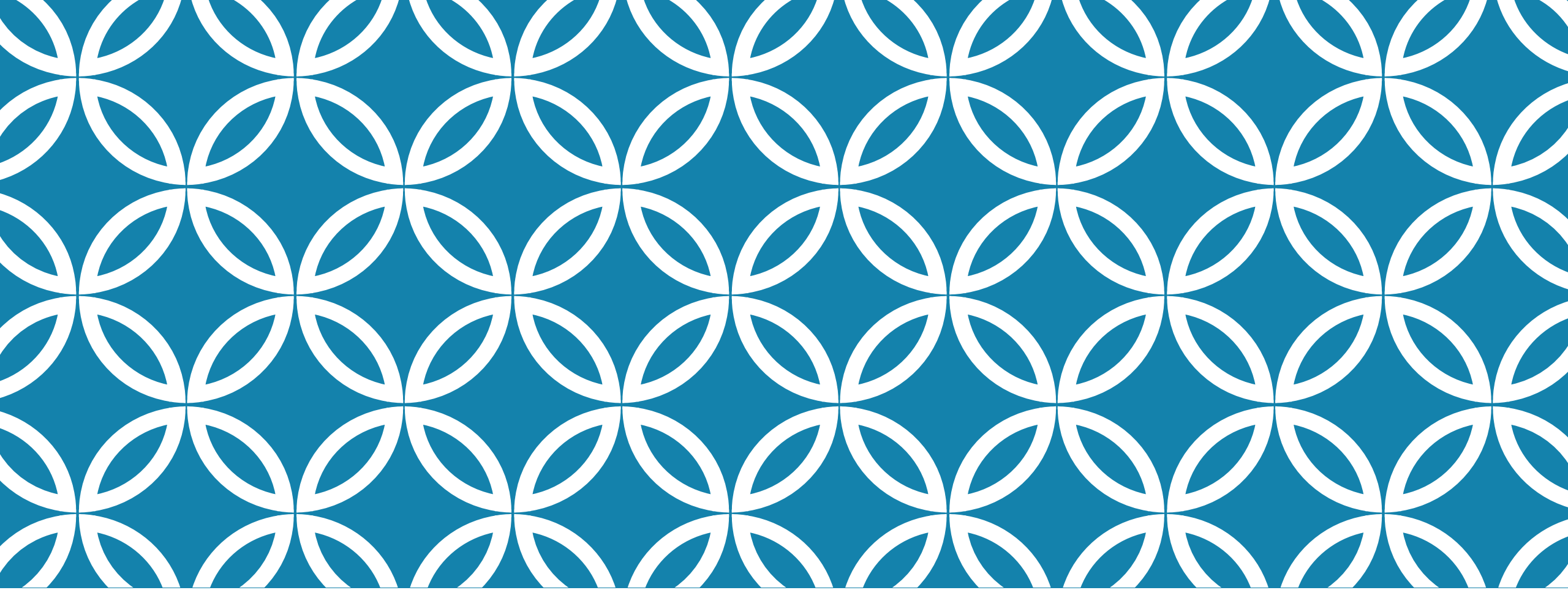
FONCTIONS MATH

Il s'agit d'un objet définissant de nombreuses constantes et fonctions mathématiques.

```
alert(Math.E);                    // la valeur de e
alert(Math.PI);                   // la valeur de pi
alert(Math.min(5,12));            // 5
alert(Math.max(23,5,7,130,12));   // 130
alert(Math.ceil(25.3));           // 26
alert(Math.floor(25.8));          //25
alert(Math.random());             // valeur aleatoire entre 0 et 1

var n = Math.floor(Math.random()*nb + min);
alert(n);                         // valeur aleatoire entre min et min+nb (exclus)
```

D'autres fonctions :

- Math.abs(x) Math.exp(x) Math.log(x)
- Math.pow(x,y) Math.sqrt(x)
- Math.sin(x) Math.cos(x) Math.tan(x)



LES STRUCTURES DE CONTRÔLE

STRUCTURES DE CONTRÔLE

Elles sont très proches de celles de langages tels que C, C++ et Java.
Pour rappel, les structures de contrôle sont de trois types :

– **Séquence** : exécution séquentielle d'une suite d'instructions séparées par un point-virgule
– **Alternative** : structure permettant un choix entre divers blocs d'instructions suivant le résultat d'un test logique
– **Boucle** : structure itérative permettant de répéter plusieurs fois le même bloc d'instructions tant qu'une condition de sortie n'est pas avérée

Il ne faut pas confondre x == y (test d'égalité) avec x = y (affectation).

On peut aussi vérifier le type d'une variable

```
var x = 4;
var y = "4";
x==y        //true
x===y       //false
```

ALTERNATIVE

L'instruction **if...else** :

```
if (rank == 1) {medaille="or";}
else if (rank == 2) {medaille="argent";}
else if (rank == 3) {medaille="bronze";}
else{medaille="élimination";}
```

L'operateur ternaire ? : permet de remplacer une instruction if...else simple.

Sa syntaxe (lorsqu'utilisée pour donner une valeur à une variable) est :
variable = condition ? expressionIf : expressionElse;

```
var civilite = (sexe == "F") ? "Madame" : "Monsieur";
```

BOUCLE

L'instruction **while** :

```
while (condition) { instruction1; instruction2; }
```

L'instruction **for** :

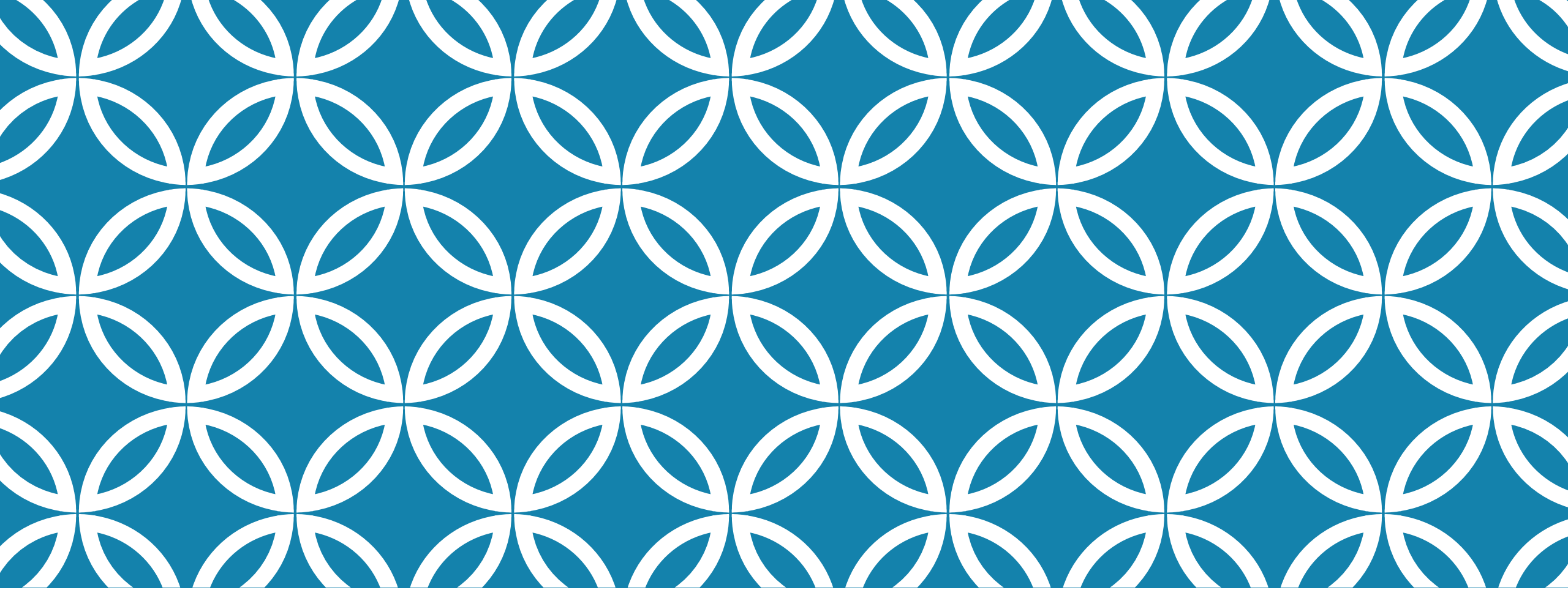
```
for (var num = 1; num <= 5; num++) alert(num);
```

L'instruction **do...while** :

```
do {instruction1; instruction2; }
while (condition);
```

L'instruction **for-in** pour les objets :

```
for (var prop in window) document.writeln(prop);
```



TYPES ET CLASSES

OBJECT

Utilisé pour stocker des données, création d'une variable (de type Object) avec new Object() ou de manière littérale (énumération entre accolades)

```
var person = new Object();
person.name = "Jordan";
person.age = 27;
var person = { name : "Jordan", age : 27 }
```

Il est possible d'utiliser les crochets pour accéder à un champ.

```
alert(person.name);

alert(person["name"]);

var field="name"; alert(person[field]);
```

ARRAY

Les tableaux peuvent contenir des données de nature différente.

```
var colors = new Array();                          // tableau vide
var colors2 = new Array(20);                       // tableau avec 20 cases
var colors3 = new Array("red","blue","green");     // 3 cases
var colors4 = ["red","blue","green"];              // notation littérale
```

Le champ **length** indique la taille d'un tableau.

```
var colors = ["red","blue","green"];

alert(colors.length);                 // 3
colors[colors.length]="black";        // nouvelle couleur
colors[99]="pink";
alert(colors.length);                 // 100
alert(colors[50]):                    // undefined
colors.length=10;                     // plus que 10 cases
```

ARRAY

De nombreuses méthodes existent sur les tableaux.

```
var colors = ["red","blue","green"];

alert(colors);                  // red,blue,green
alert(colors.join(";"));        // red;blue;green
colors.push("black");
alert(colors);                  // red,blue,green,black

var item = colors.pop();
alert(item + " " + colors);     // black red,blue,green

var item2= colors.shift();
alert(item2 + " " + colors);    // red blue,green
```

Il existe d'autres méthodes telles que **concat**, **slice** et **splice**.

RÉORDONNER LES TABLEAUX

On peut utiliser reverse et sort.

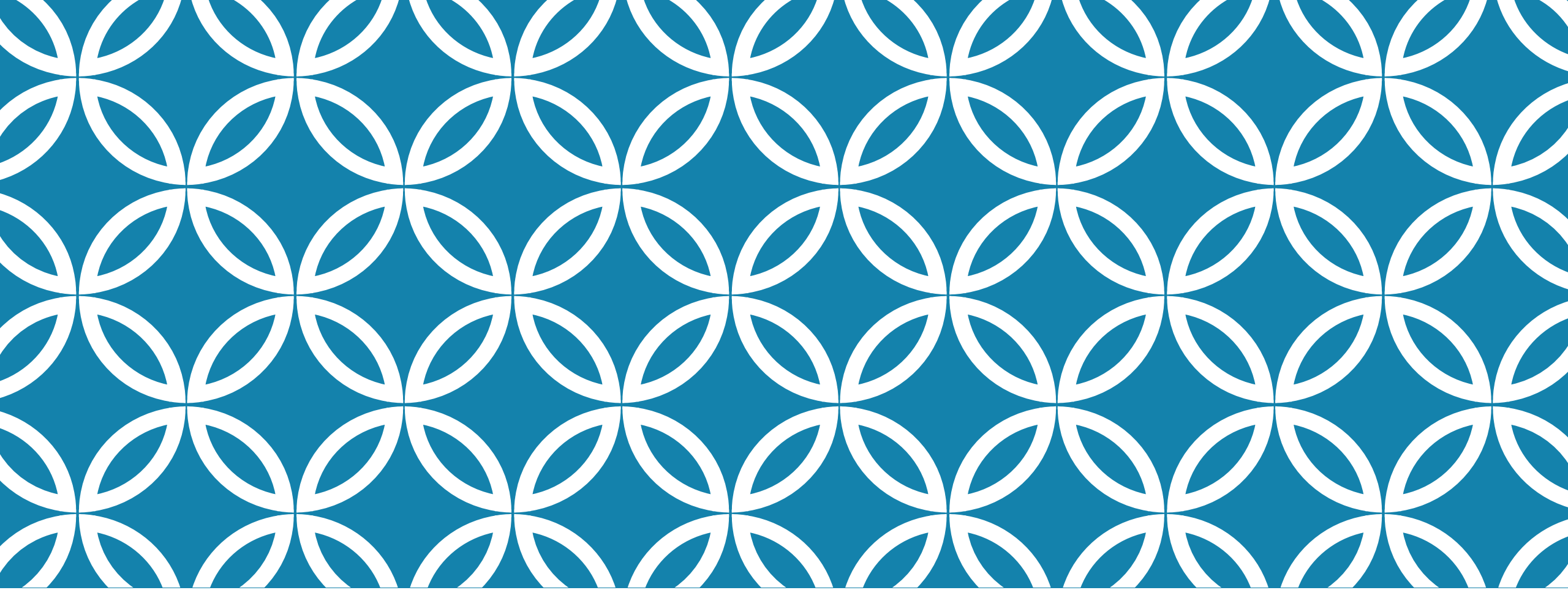
```
var values = [0, 1, 2, 3, 4];

alert(values.reverse()); // 4,3,2,1,0

values = [1, 5, 0, 15, 10];
alert(values.sort()); // 0,1,10,15,5

function compare(a, b) {return a - b; }
values = [1, 5, 0, 15, 10];

alert(values.sort(compare)); // 0,1,5,10,15
```

FONCTIONS

La syntaxe pour définir une fonction est :

```
function name(arg0, arg1, ..., argN)
{
        Traitement
}
```

Une fonction peut retourner un résultat avec l'instruction return même si rien ne l'indique au niveau de la signature de la fonction.



NAVIGATEUR

WINDOW

L'objet **window** représente la fenêtre du navigateur.

```
window.moveTo(0,0);
window.resizeTo(800,800);

var ajcWindow =window.open("https://www.ajc-formation.fr","ajc");
if (ajcWindow == null) alert("fenetre bloquee");
else {ajcWindow.close(); }
```

Le navigateur peut être configuré de manière a empêcher de modifier son emplacement, de modifier sa taille ou encore d'afficher une fenêtre pop-up.

TIMER

Il est possible de programmer l'exécution d'une méthode à un instant donné ou à des temps réguliers grâce a **setTimeout** et **setInterval**.

```
function helloWorld() { alert("hello world"); }

var id = setTimeout(helloWorld, 1000);

var num=0, max=4;
var intervalId = setInterval(incrementNumber,500);

function incrementNumber() {
        if (++num == max) clearInterval(intervalId);
        else alert(num);
}
```

Il est possible d'annuler avec clearTimeout et clearInterval

BOITES DE DIALOGUES

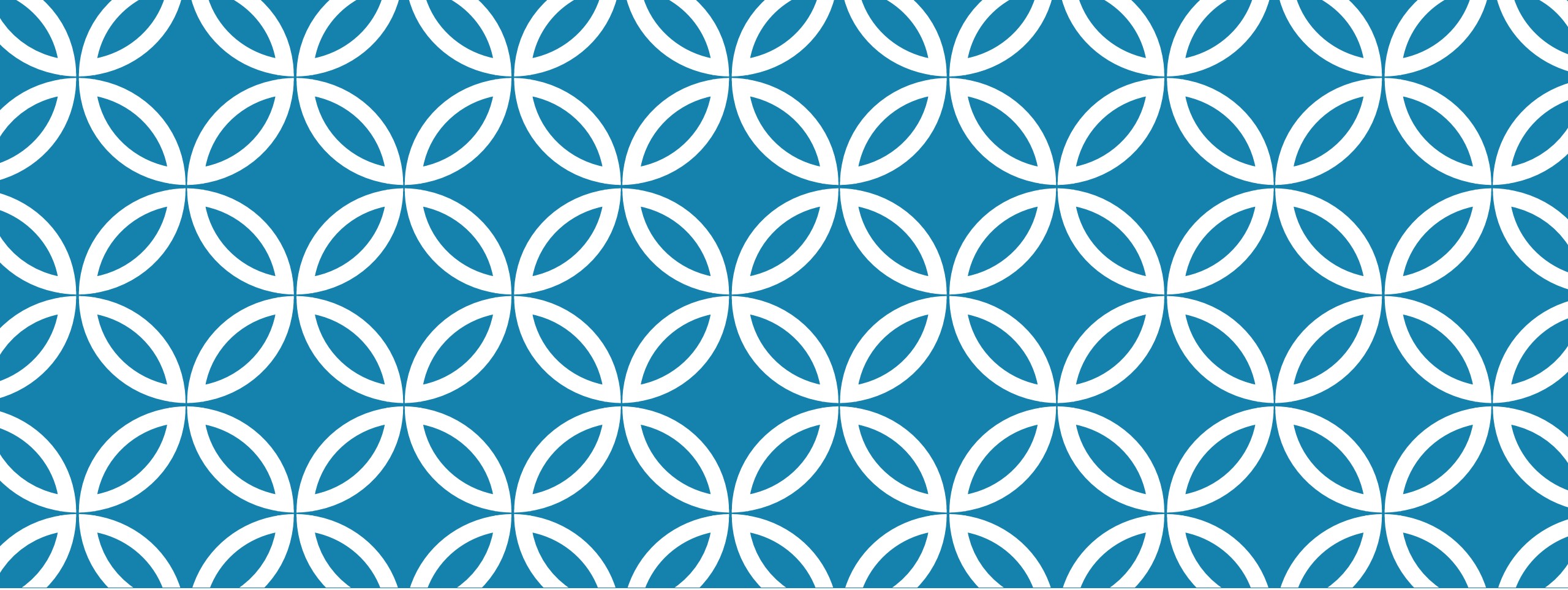
Des boites de dialogues peuvent être ouvertes en utilisant les méthodes **alert**, **confirm** et **prompt**.

```
if (confirm("Acheter ?"))
{
	alert("Merci");
}
else
{
	alert("Pas de soucis");
}

var name = prompt("Saisir votre nom ?","Jordan");
if (name != null)
{
	alert("Welcome " + name);
}
```

OBJETS DU BOM (BROWSER OBJECT MODEL)

En plus de **window**, il est possible d'utiliser les objets **location**, **navigator**, **screen** et **history**.

```
location.href="http://www.ajc-formation.fr";
location.reload();
location.port=8080;                    // change le port
alert(navigator.appName);              // nom du navigateur

alert(navigator.javaEnabled);
alert(screen.width);

window.resizeTo(screen.availWidth,screen.availHeight);

history.go(2);                         // go forward 2 pages
history.back();                        // go back one page
if (history.length == 0) alert("this is the first page");
```



DOM DOCUMENT OBJECT MODEL

DOM

DOM (Document Object Model) est une API pour manipuler les documents HTML et XML.

L'objet document représente la page HTML chargée par le navigateur.

```
var html = document.documentElement;
alert (html == document.firstChild);   // true

var body = document.body;
document.title="nouveau titre";

var url = document.URL;
var domain = document.domain;
alert(url + " " + domain);
```

MÉTHODE GETELEMENTBYID()

Cette méthode prend comme argument l'id d'un élément.

```
<head>
</head>
<body>
	<p id="p1"> liste : </p>
</body>

<script>
	var p = document.getElementById("p1");
	p.style.color="blue";
</script>
```

MÉTHODE GETELEMENTSBYTAGNAME()

Cette méthode prend comme argument le nom de balise des éléments à récupérer.

```
<body>
	<ol>
		<li>XHTML</li>
		<li>CSS</li>
	</ol>
</body>

<script>
var list = document.getElementsByTagName("li");
for (var i=0; i< list.length; i++)
{
	list[i].style.color="green" ;
}
</script>
```

(getElementsByClassName() marche de la même façon)

PROPRIÉTÉ INNERHTML

En utilisant cette propriété, il est possible de récupérer ou de modifier le contenu d'un élément HTML.

```
var div1 = document.getElementById("d1");
var div2 = document.getElementById("d2");
var div3 = document.getElementById("d3");

div2.innerHTML=div1.innerHTML;

div3.innerHTML="<strong>ora ora ora</strong>";
```

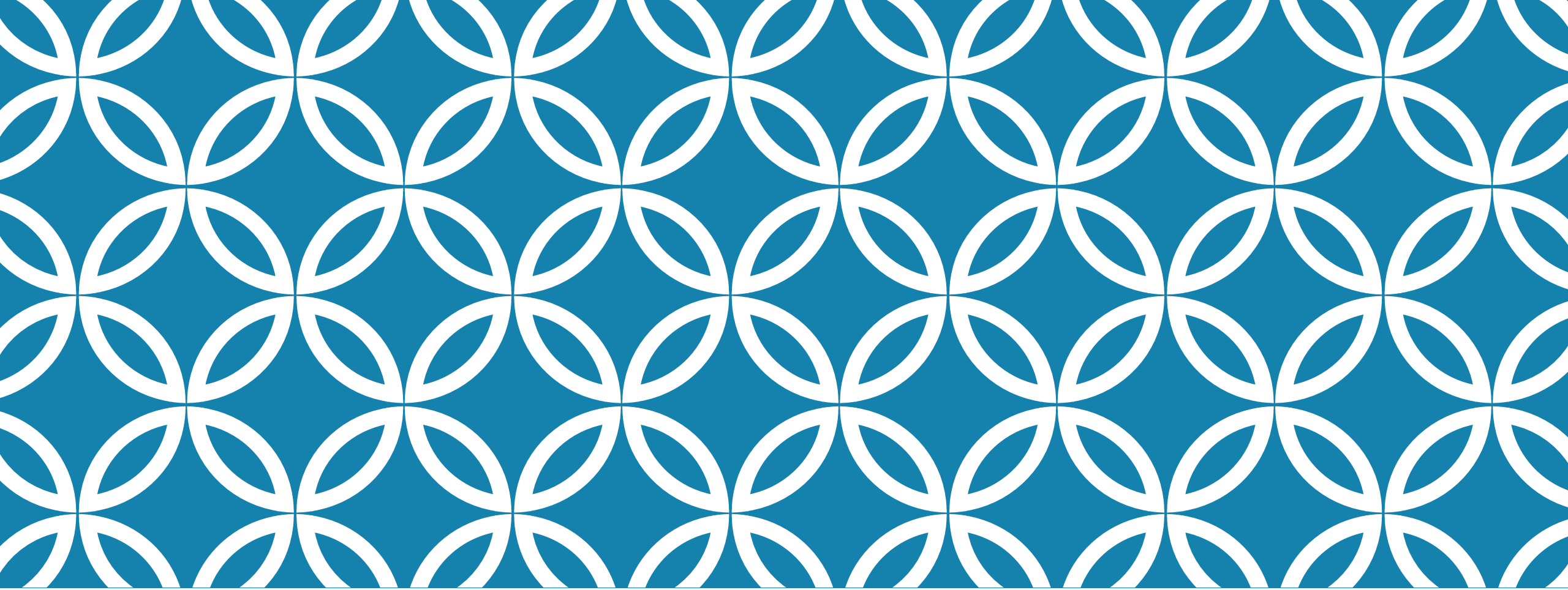
La propriété **innerText** (*textContent* pour Firefox) est similaire a **innerHTML,** mais ne traite que du texte simple.

MODIFIER LE STYLE DYNAMIQUEMENT

Tout élément HTML dispose d'un attribut **style** en JavaScript.
Les noms des propriétés CSS doivent être convertis en camel case. Par exemple :

Propriété CSS	Propriété JavaScript
background-image	style.backgroundImage
color	style.color
font-family	style.fontFamily

```
var myDiv = document.getElementById("div");
myDiv.style.backgroundColor="red";
myDiv.style.width="100px";
myDiv.style.border="1px solid border";
```



EVÈNEMENTS

EVÈNEMENTS

Un évènement est provoqué par une action de l'utilisateur ou du navigateur lui-même.

Les évènements ont des noms tels que **click**, **load** et **mouseover**.

Une fonction appelée en réponse a un évènement se nomme un écouteur (**event handler ou event listener**).

- Souvent, leur nom commence par ***on*** comme par exemple **onclick** ou **onload**.

Associer des écouteurs aux évènement possibles peut se faire de trois manières différentes:

- HTML
- DOM Level 0 (pas présenté ici)
- DOM Level 2

HTML EVENT HANDLERS

On utilise des attributs HTML pour déclarer les écouteurs.

La valeur de ces attributs est le code JavaScript à exécuter lorsque l’ évènement est produit.

```
<input type="button" value="but1" onclick="alert('clicked')" />
<input type="button" value="but2" onclick="showMessage()" />

<script type="text/javascript">
        function showMessage() { alert("hello world"); }
</script>
```

DOM LEVEL 2 EVENT HANDLERS

On utilise les propriétés des éléments pour leur associer des écouteurs.

```
<input type="button" id="but1" value="bouton 1" />

<script>
	function but1Listener() { alert("but1 cliqued"); }

	var but1 = document.getElementById("but1");

	but1.onclick=but1Listener;
</script>
```

Pour éliminer l'écouteur, on place la valeur null.
Par exemple :

```
but1.onclick=null;
```

L'OBJET EVENT

Quand un évènement se produit, toutes les informations le concernant sont enregistrées dans un objet appelé **event.**

Il est possible de récupérer cet objet sous forme de paramètre d'une fonction écouteur.

```
function but3Listener(event) {
	switch(event.type) {
		case "click" : alert("clicked at " + event.clientX + " " + event.clientY); break;
		case "mouseover" : this.style.backgroundColor="red"; break;
		case "mouseout" : this.style.backgroundColor=""; break;
	}
 }

but3.onclick=but3Listener;
but3.onmouseover=but3Listener;
but3.onmouseout=but3Listener;
```

EVÈNEMENTS SOURIS

Ce sont :

click	dblclick
mousedown	mouseup
mouseover	mouseout
mousemove	

Les propriétés utiles et accessibles à partir de l'objet **event** sont :

clientX	clientY
screenX	screenY
shiftkey	ctrlKey
altKey	metaKey

EVÈNEMENTS CLAVIER

Ce sont :

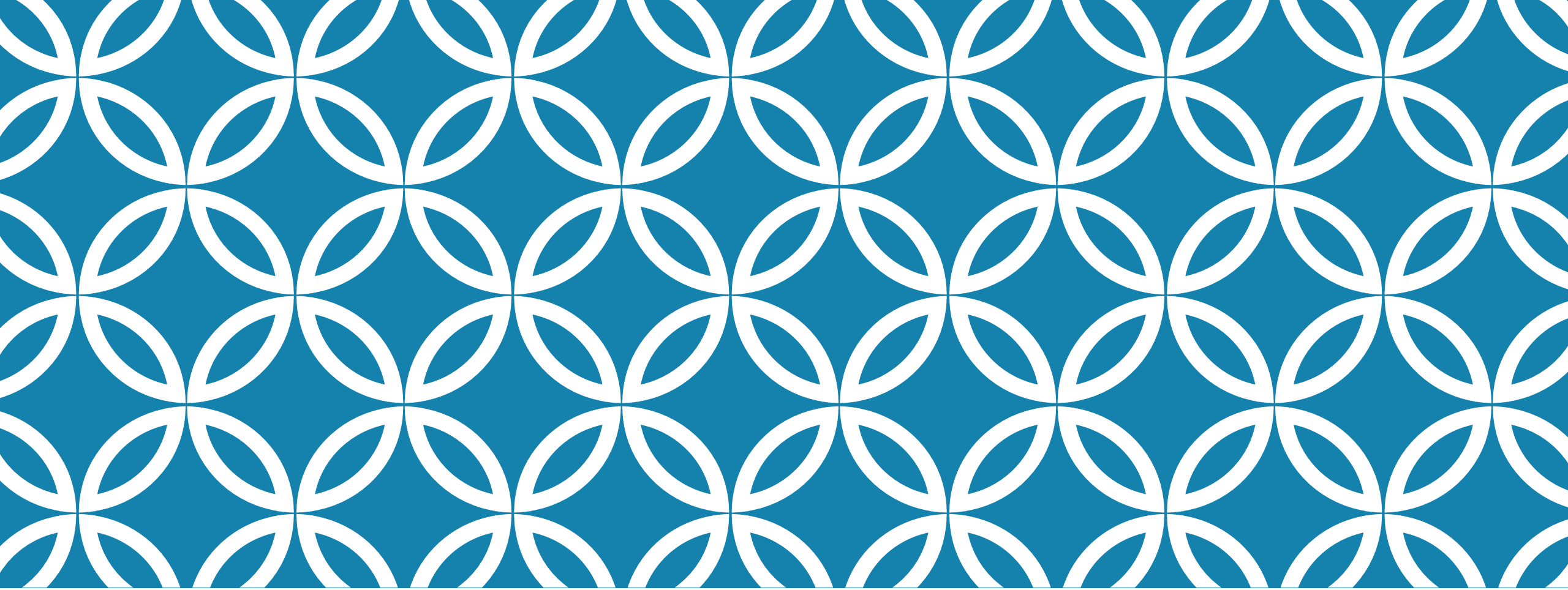
keydown	keyup
keypress	

Les propriétés utiles et accessibles à partir de l'objet **event** sont :

shiftkey	ctrlKey
altKey	metaKey
keyCode	

Exemples de valeurs pour **keyCode**:

- 40 pour Down Arrow
- 65 pour A
- 112 pour F1



BIBLIOTHÈQUE JQUERY

INTÉRÊTS DE JQUERY

jQuery est une bibliothèque qui permet d'agir sur le code HTML, CSS, JavaScript et AJAX bien plus simplement.
Il permet de manipuler les éléments mis en place en HTML ainsi qu'en CSS en utilisant des instructions simples qui donnent accès aux immenses possibilités de JavaScript et d'AJAX.

Pour l'utiliser, il suffit de l'ajouter dans nos pages web

Exemples d'améliorations :

`document.getElementById("id1);`	DEVIENT	`$("#id1");`
`document.getElementsByClassName("rouge");`	DEVIENT	`$(".rouge");`

UTLISATION DE JQUERY

```
$(document).ready(function() {Traitement});      // Le traitement se déclenche après le chargement complet de la page

$( "#secu" ).click(f1());                        // Déclenche la fonction f1 au click sur l'id secu

$( ".visite").focusin(f2());                     // Déclenche la fonction f2 lorsqu'un élément avec la classe visite est ciblé

$('#result-search').css('display','none');}.

$('#result-search').html("muda muda muda");

$('#result-search').show();

$( "li[id^='onglet-']" ).removeClass( "active" );

var type=$("#AddFormation").attr("typeForm");
```

UTLISATION D'AJAX

```
$.ajax("chemin", {
        type: "POST",
        data: {
                action:'addShop',
                idFormation: idFormation,
                type: type
        },
        success: function (resp) {
                $(#divModif).html(resp);
        }
});
```

UTLISATION D'AJAX ET DES FORMULAIRES

```
<form id='formulaire'/>
<script>
	var data = $('#formulaire').serialize()
	$.ajax("chemin.html", {
		type: "POST",
		data: data,
		success: function (resp) {
			location.reload();
		}
	});
</script>
```