

# ***Programmation Web Avancée (côté client)***

---



**Andrea G. B. Tettamanzi**

Université de Nice Sophia Antipolis

Département Informatique

[andrea.tettamanzi@unice.fr](mailto:andrea.tettamanzi@unice.fr)

*CM - Séance 05*

# **Objets standard et gestion des erreurs**

# *Plan*

- Objets intégrés
- Gestion des erreurs

# Objets intégrés

- Un certain nombre d'objets intégrés sont disponibles lors qu'un programme JavaScript est exécuté.
- L'objet global fait directement partie de l'environnement lexical du programme
  - Il contient les variables et les fonctions « globales »
- Les autres objets intégrés sont accessibles comme propriétés initiales de l'objet global
- Beaucoup des objets intégrés sont des fonctions
  - Ils peuvent être appelés avec des arguments
  - Certains sont, en outre, des constructeurs
- Les noms de ces objets s'inspirent lourdement aux noms des classes de la plate-forme Java et de leurs méthodes et attributs

## L'objet « Object »

- Permet de créer un objet de conversion pour une valeur donnée
- Par défaut, le prototype des objets créés est « null »
- Parmi ses méthodes on trouve
  - create : crée un nouvel objet avec l'objet prototype et les propriétés spécifiées
  - keys : renvoie un tableau contenant les noms de toutes les propriétés énumérables de l'objet donné
  - getPrototypeOf(o) : renvoie le prototype de l'objet o passé comme argument

## *L'objet « Function »*

- Toute fonction en JavaScript est en réalité un objet qui hérite le prototype de Function
- Comme constructeur Function crée des fonctions à run-time
- Parmi les méthodes des objets créés par Function on peut citer :
  - `apply(o [, args])` : applique l'objet fonction comme s'il était une méthode de l'objet `o` passé comme argument ; les arguments de la fonction doivent être passés comme un objet Array
  - `bind` : crée une nouvelle fonction qui, lorsqu'elle est appelée, invoque cette fonction comme si c'était une méthode de la valeur fournie
  - `call` : la même chose que `apply`, mais avec les arguments passés un par un au lieu que comme un objet Array

## *L'objet « Array »*

- Constructeur de tableaux
- Conteneurs de haut niveau, dont le comportement et les caractéristiques les rapprochent à des listes
- Un tableau peut être créé soit par une expression littérale,
  - [élément 0 , élément 1, . . . , élément n-1 ]
- soit par une invocation du constructeur Array :
  - new Array(élément 0 , élément 1 , . . . , élément n-1 )
  - new Array(n)
- La propriété length d'un tableau contient toujours sa taille

## L'objet « String »

- Constructeur de chaînes de caractères
- La propriété length d'une chaîne de caractères contient sa taille
- Des chaînes de caractères peuvent être comparées en utilisant les opérateurs de comparaison <, <=, ==, >=, >, et !=
- L'opérateur de concaténation est +
- De nombreuses méthodes utilitaires font partie de cet objet :
  - charAt, concat, contains, endsWith, indexOf, match, replace, search, slice, split, substr, trim, toLowerCase, toUpperCase, ...

## *L'objet « Math »*

- Un objet qui possède des propriétés et des méthodes qui mettent à disposition du programmeur des constantes et des fonctions mathématiques.
- Math n'est pas un constructeur
- Toutes les propriétés et méthodes de Math sont statiques

## *L'objet « Date »*

- Crée des objets qui représentent des dates et des temps
- Permet de manipuler des dates et des temps
  - `new Date()`
  - `new Date(u)` : avec l'heure POSIX
  - `new Date(chaîne)`
  - `new Date(A, M , J [, h, m, s, ms])`
- L'objet `Date` fournit trois méthode d'utilité
  - `now` : renvoie l'heure POSIX actuelle
  - `parse` : analyse une date en format chaîne de caractères
  - `UTC` : prend les mêmes arguments de le forme la plus longue du constructeur et renvoie l'objet `Date` correspondant.

# Gestion des erreurs

- Le mécanisme de gestion repose sur trois ingrédients :
  - un ou plusieurs routines de traitement d'exceptions (les handlers) ;
  - un mécanisme de signalement d'exceptions ;
  - un mécanisme qui permet d'associer les exceptions à leurs handlers.
- En JavaScript, le mécanisme d'association des exceptions à leurs handlers est fourni par la structure `try ... catch ... finally`
- Les routines de traitement d'exception sont contenues dans les clauses « `catch` » de cette construction
- Le signalement des exception se fait à l'aide de l'opérateur « `throw` » et d'objets créés par le constructeur `Error`.

*Merci de votre attention*

