

Université
Nice
Sophia Antipolis

Programmation Web - Client riche

Gaëtan Rey
Gaetan.Rey@unice.fr
DUT Informatique – 2016

CC BY NC ND

Université
Nice
Sophia Antipolis

Objectifs

- ▶ Introduction / Rappel sur le web
- ▶ Navigateurs et DOM
- ▶ JavaScript
- ▶ Requêtes asynchrones et formats d'échanges de données
- ▶ Exemples de bibliothèques et autres langages

Janvier 2016 Gaëtan Rey – Université Nice Sophia Antipolis 2

Université
Nice
Sophia Antipolis

EXEMPLES DE BIBLIOTHÈQUES ET AUTRES LANGAGES

Janvier 2016 Gaëtan Rey – Université Nice Sophia Antipolis 84

Université
Nice
Sophia Antipolis

Les bibliothèques JavaScript

- ▶ Objectifs
 - ▶ Faciliter l'usage du JavaScript
 - ▶ Faire abstraction
 - ▶ des différentes implémentations du langage et
 - ▶ des différents navigateurs
- ▶ Elles se doivent d'être pérenne, donc d'avoir
 - ▶ une documentation de qualité
 - ▶ une large communauté offrant un support de qualité
 - ▶ un ratio performances/contraintes favorable
 - ▶ une mise en œuvre aisée
 - ▶ l'adhésion d'acteurs de premier plan
 - ▶ une équipe pour porter le projet qui doit être dynamique et en évolution permanente

Janvier 2016 Gaëtan Rey – Université Nice Sophia Antipolis 85

Université
Nice
Sophia Antipolis

jQuery : Présentation

- ▶ [jQuery](#) est une bibliothèque JavaScript **libre** et **multi-plateforme** créée pour **faciliter l'écriture** de scripts côté client dans le code HTML des pages web. La première version est lancée en janvier 2006 par John Resig. [[Wikipédia](#)]
- ▶ Principales fonctionnalités :
 - ▶ Parcours et modification du DOM dont le CSS
 - ▶ Gestion des événements
 - ▶ Ajout d'effets visuels et animations
 - ▶ Prise en charge de la gestion asynchrone et des formats de données JSON et XML

Janvier 2016 Gaëtan Rey – Université Nice Sophia Antipolis 86

Université
Nice
Sophia Antipolis

jQuery : les pré-requis

- ▶ Pour utiliser efficacement jQuery, il est conseillé :
 - ▶ De connaître les langages CSS, HTML, JavaScript et un langage serveur pour la partie serveur des transactions AJAX.
 - ▶ De maîtriser, en JavaScript, la notation JSON (JavaScript Object Notation).
 - ▶ D'avoir un bon niveau en programmation fonctionnelle, en programmation par objets et en programmation événementielle.

[[Développez.com](#)]

Janvier 2016 Gaëtan Rey – Université Nice Sophia Antipolis 87

Version de jQuery

- jQuery est distribuée
 - sous 2 formes
 - jquery.js : la version non compressée (développement)
 - jquery.min.js : la version compressée (production)
 - Sous 2 familles de version
 - 1.x : version recommandée
 - 2.x : version sans le support d'IE 6,7 et 8. Non recommandée actuellement

Pour ce cours, la version 1.11.2 a été utilisée comme référence

Janvier 2016 Gaëtan Rey - Université Nice Sophia Antipolis 88

Installer jQuery

- Version locale
 - Déposer le fichier js sur votre serveur
 - Ajouter la ligne suivante dans vos pages html


```
<script charset="utf-8" src="cheminLocal/jquery-X.Y.Z.min.js"></script>
```
- Version distante
 - Ajouter une des ligne suivantes dans vos pages html


```
<script charset="utf-8" src="http://code.jquery.com/jquery-X.Y.Z.min.js"></script>
<script charset="utf-8" src="http://ajax.aspnetcdn.com/ajax/jquery/jquery-X.Y.Z.min.js"></script>
<script charset="utf-8" src="http://ajax.googleapis.com/ajax/libs/jquery/X.Y.Z/jquery.min.js"></script>
```

Janvier 2016 Gaëtan Rey - Université Nice Sophia Antipolis 89

L'opérateur jQuery

- jQuery(...)** : Retourne un objet jQuery contenant une collection d'éléments du DOM
 - Peut être abrégé **\$(...)**
- jQuery(selecteur[, contexte])** : l'objet retourné contient l'ensemble des éléments du *contexte* qui correspondent au *selecteur* CSS
- jQuery(element)** : l'objet retourné contient l'élément passé en paramètre
- jQuery()** : l'objet retourné contient un ensemble vide

Janvier 2016 Gaëtan Rey - Université Nice Sophia Antipolis 90

L'opérateur jQuery

- jQuery(...)** : Retourne un objet jQuery contenant une collection d'éléments du DOM
- jQuery(TableauElements)** : l'objet retourné contient les éléments du tableau passé en paramètre
- jQuery(objetjQuery)** : l'objet retourné est un clone de l'objet passé en paramètre
- jQuery(objetJavascript)** : l'objet retourné est une « version » jQuery de l'objet passé en paramètre. Mais les actions utilisables sur ce dernier sont limitées

Janvier 2016 Gaëtan Rey - Université Nice Sophia Antipolis 91

Exemples

```
$( "div" ); // sélection des balises DIV
$( ".toto" ); // sélection des éléments de classe toto
$( "input[nom='rey']" ); // sélection des balises INPUT dont la valeur de l'attribut nom vaut rey
$( "input[nom*=rey]" ); // sélection des balises INPUT dont la valeur de l'attribut nom contient la sous chaîne rey
$( "input[nom~='rey']" ); // sélection des balises INPUT dont la valeur de l'attribut nom contient le mot rey
$( "input[nom$='rey']" ); //sélection des balises INPUT dont la valeur de l'attribut nom se termine par rey
$( "div.toto" ).click( function() { // abonnement au click pour les balises DIV de classe toto
  $( "p", this ).addClass( "titi" ); // ajout de la classe titi aux balises P de l'élément // précédemment sélectionné. $( "span", this ) est équivalent à $( this ).find( "span" )
});
$( "div.toto" ).click( function() {
  $( this ).slideUp(); // cache l'élément avec un mouvement de glissement
});
$.post( "rss.xml", function( data ) { // charge les données depuis le serveur via une requête POST
  var $child = $( data ).find( "item" ); // sélectionne les balises ITEM dans les données
});
```

Janvier 2016 Gaëtan Rey - Université Nice Sophia Antipolis 92

ready ?

- Attention, comme en JavaScript, il ne faut pas parcourir ou modifier le DOM avant que celui-ci ne soit opérationnel
- Pour détecter cela en jQuery, il y a l'évènement **jQuery().ready()**

```
$(document).ready( function(){
  ....
});
```
- Attention, cela est globalement incompatible avec l'évènement **onload** utilisé habituellement

Janvier 2016 Gaëtan Rey - Université Nice Sophia Antipolis 93

Les évènements en général

- En plus de `ready()`, on trouve de nombreux autres évènements commun avec ceux de JavaScript dont
 - `blur()`, `change()`, `click()`, `dblclick()`, `focus()`, `focusin()`, `focusout()`, `hover()`, `keydown()`, `keypress()`, `keyup()`, `mousedown()`, `mouseenter()`, `mouseleave()`, `mousemove()`, `mouseout()`, `mouseover()`, `mouseup()`, `ready()`, `resize()`, `scroll()`, `select()`, `submit()`, ...
- On trouve en plus quelques fonctions
 - `off()`, `on()`, `one()`, `trigger()`, ...

Janvier 2016 Gaëtan Rey - Université Nice Sophia Antipolis 94

Exemples

```
$(document).keydown(function() { // abonnement à l'évènement keydownm l'objet document
  ...
});

$("#toto").on("click", function( event ) { // ajoute l'évènement click à la balise d'id toto
  ...
  $(this).off( event ); // supprime à l'élément courant (this) son abonnement à l'évènement
  // courant (event)
});

// equivalent à

$("#toto").one( "click", function() { // ajoute l'évènement click à la balise d'id toto pour une
  // seule utilisation
});

$("#toto").trigger( "click" ); // déclenche la méthode associée à l'évènement click sur l'objet
// sélectionné
```

Janvier 2016 Gaëtan Rey - Université Nice Sophia Antipolis 95

Manipulation d'une Collection

- `.each(fonction)`** : parcourt l'objet jQuery et appelle la fonction sur l'ensemble des éléments.


```
$( "li" ).each( function(index, element) { // index : le rang de l'élément
  console.log( index + " : " + element.text() ); // element: l'élément, ici l'objet LI
});

$( "li" ).each( function( index ) { // le context est mis à jour à chaque appel
  console.log( index + " : " + $( this ).text() ); // ici, this est bien l'élément courant
});
```
- Il existe aussi de nombreuses fonctions qui s'appliquent directement sur l'ensemble des éléments de l'objet jQuery

Janvier 2016 Gaëtan Rey - Université Nice Sophia Antipolis 96

AJAX (1)

- `jQuery.ajax()`**

```
// exemple 1 : sauvegarde des données sur le serveur et notification de succès
// exemple 2 : tester les codes de réponse HTTP
$.ajax({
  ...
  statusCode: {
    404: function() {
      ... // ajouter aussi le code de traitement
    }
  }
  ...
});
```

Janvier 2016 Gaëtan Rey - Université Nice Sophia Antipolis 97

AJAX (2)

- `jQuery.get(url [, data] [, success] [, dataType])`**
 - Version simplifiée pour accéder émettre des requêtes HTTP Get

```
// exemple 1 : exemple compact qui ne gère que le succès
// exemple 2 : exemple avec fonction de succès et type de données
$.get( "file.php", function( data ) {
  $( "body" )
    .append( "Nom: " + data.nom ) //
    .append( "Lieu: " + data.lieu );
}, "json");
```

Janvier 2016 Gaëtan Rey - Université Nice Sophia Antipolis 98

AJAX (3)

- `jQuery.post(url [, data] [, success] [, dataType])`**
 - Version simplifiée pour accéder émettre des requêtes HTTP Post

```
// exemple 1 : exemple compact qui ne gère que le succès
// exemple 2 : exemple avec des données envoyées au serveur
$.post( "file.php", { nom: "Rey", lieu: "Nice" } )
  .done(function( data ) {
    ...
  });
```

Janvier 2016 Gaëtan Rey - Université Nice Sophia Antipolis 99

Les effets (1)

- Un des gros points fort de jQuery était de proposer un ensemble d'animations bien avant CSS3
- `.animate(propriétés [, duration] [, easing] [, complete])`
- `.animate(propriétés, options)`
 - déclenche une animation personnalisée sur des éléments CSS.

```
// exemple 2 : exemple compact qui ne gère que le succès
$( "#id1" ).click(function() {
  $( "#id2" ).animate({ // liste de 3 propriétés à modifier dans l'animation
    opacity: 0.25, // modifiée uniquement par la 1ere animation
    left: "+=50", // animée à chaque itération
    height: "toggle" // réduit la taille à 0 si l'objet est visible, l'inverse la fois suivante
  }, 5000, function() {
    // le code à déclencher à la fin de l'animation
  });
});
```

Janvier 2016 Gaëtan Rey - Université Nice Sophia Antipolis 101

Les effets (2)

- Affiche les éléments via un fondu (vers opaque)
- `.fadeIn([duration] [, complete])`
- `.fadeIn(options)`

```
$( document.body ).click(function() {
  $( "div: hidden: first" ).fadeIn( "slow" ); // affiche doucement le premier DIV caché
});

...
<p>ici se trouve la solution à votre problème <div><span>Solution ?</span></div></p>
...
$( "span" ).click(function() {
  $( "div" ).fadeOut( 3000, function() {
    $( "span" ).fadeOut( 100 );
  });
});
```

Janvier 2016 Gaëtan Rey - Université Nice Sophia Antipolis 101

Les effets (3)

- Cache les éléments
 - `.hide()` : sans animation
 - `.hide([duration] [, complete])`
 - `.hide(options)`
 - `.hide(duration [, easing] [, complete])`
- Affiche les éléments
 - `.show()` : sans animation
 - `.show([duration] [, complete])`
 - `.show(options)`
 - `.show(duration [, easing] [, complete])`

Janvier 2016 Gaëtan Rey - Université Nice Sophia Antipolis 102

Les effets (4)

- Cache les éléments en les tassant (vers le haut)
 - `.slideUp([duration] [, complete])`
 - `.slideUp(options)`
 - `.slideUp([duration] [, easing] [, complete])`
- Affiche les éléments en les étirant (vers le bas)
 - `.slideDown([duration] [, complete])`
 - `.slideDown(options)`
 - `.slideDown([duration] [, easing] [, complete])`

Janvier 2016 Gaëtan Rey - Université Nice Sophia Antipolis 103

Les effets (5)

- Ne pas oublier les versions « Toggle » de chaque effet
 - `.toggle()`
 - `.slideToggle()`
 - `.fadeToggle()`
- Il cache l'élément s'il est visible et l'affiche s'il est caché
- Idéal pour faire des effets de type menu

Janvier 2016 Gaëtan Rey - Université Nice Sophia Antipolis 104

Quelques grands classiques

- Est-ce qu'un élément est de telle classe ?
 - Avec la fonction `is()` : si votre fonction de sélection retourne une liste, `is()` répondra vrai si *au moins un élément* de la liste possède votre classe.

```
if ( $(monElement).is( ".maClasse" ) ) { // is() peut aussi tester d'autres expressions
  ...
}
```
- Avec la fonction `hasClass()` : si votre fonction de sélection retourne une liste, `hasClass()` répondra vrai si *tous les éléments* de la liste possèdent votre classe.

```
if ( $(monElement).hasClass( "maClasse" ) ) {
  ...
}
```

Janvier 2016 Gaëtan Rey - Université Nice Sophia Antipolis 105

Université Nice Sophia Antipolis

Quelques grands classiques

- Est-ce qu'un élément est visible ou caché ?
 - Utilisation des sélecteurs ou via des filtres
 - :hidden** : sélectionne tous les éléments qui sont cachés
 - :visible** : sélectionne tous les éléments qui sont visibles

```

$("div:hidden").show("slow"); //affiche l'élément

$("div").filter(":hidden").show("slow"); //préférable pour des questions de
// performances

```

Janvier 2016 Gaëtan Rey - Université Nice Sophia Antipolis 106

Université Nice Sophia Antipolis

Quelques grands classiques

- Est-ce qu'un élément est visible ou caché ?
 - Depuis jQuery 1.3.2
 - :visible** : si la propriété *offsetWidth* ou la propriété *offsetHeight* est supérieure à 0
 - :hidden** : c'est plus compliqué car il y a plusieurs possibilité
 - Propriété *display* a pour valeur *none*
 - C'est un élément de formulaire avec *type="hidden"*
 - L'élément a une largeur et une hauteur à 0
 - Un élément ancêtre est caché, l'élément n'est pas affiché sur la page
 - Les éléments avec *visibility: hidden* ou *opacity: 0* sont visibles
 - Les éléments qui ne sont pas dans le document sont visibles
 - La visibilité des éléments peut varier durant les animations

Janvier 2016 Gaëtan Rey - Université Nice Sophia Antipolis 107

Université Nice Sophia Antipolis

Quelques grands classiques

- Surligner des mots identiques

```

$(document).ready(function() {
  var surligner = function(texte, contexte){
    var regExp = new RegExp(texte, "gm");
    $(contexte).children().each(function(i, item){
      $(item).html(
        $(item).html().replace(regExp, "<span class='select'>" + texte + "</span>")
      );
    });
  }
  $("#buttonSearch").click(function(){
    var texte = $("#toSearch").val();
    surligner(texte, document.body);
  });
});

```

Janvier 2016 Gaëtan Rey - Université Nice Sophia Antipolis 108

Université Nice Sophia Antipolis

TypeScript

- Langage de programmation développé par Microsoft
 - libre et open-source
- Buts
 - d'améliorer la production de code JavaScript
 - de sécuriser la production de code JavaScript
- C'est un sur-ensemble de JavaScript
 - Tout code JavaScript correct peut être utilisé avec TypeScript
- Le code TypeScript est transcompilé en JavaScript,
 - Il peut être interprété par les moteurs JavaScript standards

Janvier 2016 Gaëtan Rey - Université Nice Sophia Antipolis 109

Université Nice Sophia Antipolis

TypeScript

- Il permet
 - un typage statique optionnel des variables et des fonctions,
 - la création de classes et d'interfaces,
 - l'import de modules,
- Tout en conservant l'approche non-contraignante de JavaScript
- Il supporte la spécification ECMAScript 6

[Wikipédia]

Janvier 2016 Gaëtan Rey - Université Nice Sophia Antipolis 110

Université Nice Sophia Antipolis

Quelques liens

- Site officiel : <http://www.typescriptlang.org>
- Vidéo de présentation
- Tutoriels :
 - developpez.com
 - [openclassrooms](http://openclassrooms.com)
 - [pluralsight](http://pluralsight.com)

Janvier 2016 Gaëtan Rey - Université Nice Sophia Antipolis 111

Typage statique

- Typage explicite


```
var valeur: number; //entier ou
var texte: string;
var estRouge: boolean;
var autre: any; // type par défaut
```
- Typage d'une fonction


```
function lancerDe(nbFace: number): number
{
    return Math.floor(Math.random() * nbFace);
}
```
- Typage implicite


```
var valeur = 10.45; // number
var texte = "Bonjour !"; // string
var estRouge = true; // boolean
var autre = null; // any
```
- ```
function fin(texte: string): string
{
 return texte.substring(0, texte.length-1);
}
```

Janvier 2016 Gaëtan Rey - Université Nice Sophia Antipolis 112

## La notion de classe

- Elle anticipe la prochaine évolution de JavaScript (ECMAScript 6)
 

```
class Vehicule{
 modele: string

 constructor (modele: string){
 this.modele = modele;
 }

 nbRoues(): number {
 return 4;
 }
}
```
- ```
class Moto extends Vehicule {
    modele: string
    type: string

    constructor (modele: string, type: string){
        super(modele);
        this.type = type;
    }

    nbRoues(): number {
        return 2;
    }
}
```

Janvier 2016 Gaëtan Rey - Université Nice Sophia Antipolis 113

Interfaces

- Contrat/signature que doit respecter l'implémentant
- Permet de définir des types d'objets

```
interface MonInterface {
    // Création d'une signature de variable
    maVariable: string;
    // Création d'une signature de méthode
    maMethode( parametre: string): void;
}

class MaClasse implements MonInterface {
    maVariable: string;
    maMethode( parametre: string): void {
        // Contenu de la méthode
    }
}
```

```
interface MonInterface {
    monText: string;
    maVal: number;
    maMet ( p: string, b?: bool): string;
}

function maFonction (x: MonInterface) {
    return x.maMet("gaetan");
}
```

Janvier 2016 Gaëtan Rey - Université Nice Sophia Antipolis 114

Modules (1)

- Référencement de fichiers
 - Chaque fichier .ts est implicitement un module
 - Notion proche des #include de C/C++

```
/// <reference path="module.ts"/>
```
- Module explicite
 - Permet de définir des espaces de nommage
 - Mot clé : export indique que l'élément est visible à l'extérieur du module

Janvier 2016 Gaëtan Rey - Université Nice Sophia Antipolis 115

Modules (2)

```
module MonModule {
    var t = "hello";
    export function maFonc(): string { ... }
    export interface MonInter {
        a: number;
    }
    export class MaClass implements MonInter {
        a: number;
    }
}

console.log(MonModule.maFonc()); // affiche le retour de la fonction
console.log(MonModule.t); // erreur, t n'est pas exporté
```

Janvier 2016 Gaëtan Rey - Université Nice Sophia Antipolis 116

TypeScript, c'est aussi

- Des types générique (pour les fonctions ou les classes)


```
function maFonction<T>(parametre: T) {
    ...
}
```
- Des énumérations


```
enum Couleur {Blue, Rouge, Vert};
```
- Des unions


```
function maFonction(monParametre: string | string[]): void {
    if (typeof monParametre === "string") {
        // Traitement de ma chaîne de caractère
    } else {
        // Traitement de mon tableau
    }
}
```

Janvier 2016 Gaëtan Rey - Université Nice Sophia Antipolis 117

Université Nice Sophia Antipolis

Outils de développement

- ▶ Le compilateur tsc (utilisable via node.js)
 - ▶ Transforme le code TypeScript en JavaScript
- ▶ Des environnements de développement
 - ▶ Visual Studio (version 2013)
 - ▶ WebStorm
 - ▶ Eclipse (TypEcs)
- ▶ Un site de test : <http://typescript.io>

Janvier 2016 Gaëtan Rey - Université Nice Sophia Antipolis 118

Université Nice Sophia Antipolis

Le minimum à connaître

- ▶ La notion de bibliothèque
- ▶ Les principales fonctions de jQuery
- ▶ Les principales fonctions de TypeScript

Janvier 2016 Gaëtan Rey - Université Nice Sophia Antipolis 119

Université Nice Sophia Antipolis

Quelques liens utiles

- ▶ Google Hosted Libraries
 - ▶ <https://developers.google.com/speed/libraries/>
- ▶ Microsoft Ajax Content Delivery Network
 - ▶ http://www.asp.net/ajax/cdn#jQuery_Releases_on_the_CDN_0
- ▶ Mozilla Developer Network
 - ▶ <https://developer.mozilla.org/fr/>
- ▶ Developpez.com
 - ▶ <http://web.developpez.com>

Janvier 2016 Gaëtan Rey - Université Nice Sophia Antipolis 120

Université Nice Sophia Antipolis

Bibliographie / Remerciement

- ▶ Merci
 - ▶ A la communauté Developpez.com
 - ▶ A la communauté Wikipédia
- ▶ Cours sur JavaScript utilisés pour monter ce module
 - ▶ Introduction au JavaScript de Serge P.
 - ▶ Développement Web : « Zone Grand Débutant » de Guillaume Rossolini
 - ▶ Cours de JavaScript de Jacques Guizol
 - ▶ Apprendre le JavaScript de Van Lancker Luc
 - ▶ Syntaxe JavaScript de [Wikipédia]
 - ▶ JavaScript sur le Mozilla Developer Network
- ▶ Autres supports utilisés
 - ▶ Comment fonctionnent les navigateurs

Janvier 2016 Gaëtan Rey - Université Nice Sophia Antipolis 121