jQuery, communication AJAX

- Dialogue client/serveur, simplification ...
  - ...et amélioration des requêtes XMLHTTPRequest
  - ...du traitement du flux réponse XML et JSON
- Une méthode simple de chargement : `.load()`
- Des fonctions globales de communication :
  - `$.get()`, `$.post()` et `$.getJSON()` sont des raccourcis de `$.ajax()`, la + complète
  - envoi/chargement de données en GET ou POST
  - fonction de rappel pour traiter le résultat reçu

$('#id').load()

- Charge des données par HTTP GET ou POST en injectant directement le résultat en DOM
  - $.load(url, [data], [complete])

    url : l'url du serveur à interroger

    data : les paramètres éventuels à envoyer (utilise alors la méthode POST)

    complete : la fonction de rappel (dans tous les cas)

```
$.ajax({
  type: 'GET'
  url: url,
  data: data,
  success: function(result){$('#id').html(result)},
  dataType: 'html',
  complete : function() {...}
});
```

jQuery.get()

- Dialogue client/serveur par HTTP GET
  - $.get(url, [data], [success], [dataType])

    url : l'url du serveur à interroger

    data : les paramètres éventuels à envoyer

    success : la fonction de rappel (en cas de succès, rien ne se passe en erreur)

    dataType : le type des données reçues (xml, json, script, html)

```
$.ajax({
  url: url,
  data: data,
  success: function(result) {...},
  dataType: dataType
});
```

jQuery.post()

- Dialogue client/serveur par HTTP POST
  - $.post(url, [data], [success], [dataType])

    url : l'url du serveur à interroger

    data : les paramètres éventuels à envoyer

    success : la fonction de rappel (en cas de succès, rien ne se passe en erreur)

    dataType : le type des données reçues (xml, json, script, html)

```
$.ajax({
  type: 'POST',
  url: url,
  data: data,
  success: function(result) {...},
  dataType: dataType
});
```

jQuery.getJSON()

- Dialogue client/serveur par HTTP GET impliquant un flux JSON en retour
  - $.getJSON(url, [data], [success])

    url : l'url du serveur à interroger

    data : les paramètres éventuels à envoyer

    success : la fonction de rappel (en cas de succès, rien ne se passe en erreur)

```
$.ajax({
  url: url,
  data: data,
  success: function(result) {...},
  dataType: 'json'
});
```

jQuery.ajax()

- `jQuery.ajax(url,[settings]) | .ajax([settings])`
  - les paramètres sont une mappe clé/valeur

    http://api.jquery.com/jQuery.ajax/
- Offre en plus (et entre autres) :
  - gestion de la mise en cache (`cache`)
  - différentes fonctions de rappel
    - réussite de la requête (`success`)
    - terminaison sur erreur (`error`)
    - dans tous les cas (`complete`)
  - gestion d'authentification (`username`, `password`)
  - contexte d'exécution (`context`)

Choisir le format des données

- Fragment HTML :
  - **Simplicité**, sans fonction de rappel ni parcours des résultats
  - Couplage fort avec le conteneur cible
- Format JSON :
  - Structure qui facilite la **réutilisation**
  - Analyse/traitement du résultat **très rapide avec JavaScript**
- Format XML :
  - Structure **réutilisable** et format roi de **l'interopérabilité**
  - Analyse/traitement du résultat un peu plus lent
- Format personnalisé : cf. Converters
  - http://api.jquery.com/extending-ajax/#Converters

Passer des données au serveur

- En général :
  - GET : pour le chargement de données
  - POST : pour l'envoi de données
- Toutes les fonctions jQuery permettent l'envoi :
  - pour un résultat dynamique selon paramètres
  - pour un transfert de données à sauvegarder
- Rappel sur les limites en GET
  - http://www.w3.org/Protocols/rfc2616/rfc2616-sec3.html#sec3.2.1
  - http://www.boutell.com/newfaq/misc/urllength.html

  => privilégier la méthode POST quand il le faut

Passer des données au serveur

- Par le paramètre/clé data :
  - une mappe clé/valeur que jQuery convertit en chaîne de requête, par exemple :

```
$('userInfo').load('users.php',
                   {'name': formName, 'date': formDate} );
```

-> `users.php&`**`name=Torvalds&date=1969-12-28`**

- Pratique : la sérialisation de formulaire
  - remplacer le mécanisme normal de soumission par une requête AJAX
  - `.serialize()` convertit des éléments de DOM en une chaîne de requête : http://api.jquery.com/serialize/