FAQ Ext JS/Sencha

Date de publication :

Dernière mise à jour :

Cette FAQ a été réalisée pour répondre aux questions les plus fréquemment posées sur le forum **Ext JS/Sencha**.

Nous tenons à souligner que cette FAQ ne garantit en aucun cas que les informations qu'elle contient sont correctes ; les auteurs font le maximum, mais l'erreur est humaine. Si vous trouvez une erreur, ou que vous souhaitez devenir rédacteur, lisez Comment participer à cette FAQ ?.

Sur ce, on vous souhaite une bonne lecture.

Ont contribué à cette FAQ :

L'équipe JavaScript (JavaScript) -
vermine - sekaijin - JulienFio - abraxis -

Sommaire > Informations générales

Comment bien utiliser cette FAQ ?

Auteurs : sekaijin , vermine ,

Le but
Cette FAQ a été conçue pour être la plus simple possible d'utilisation. Elle tente d'apporter des réponses simples et complètes aux questions auxquelles ont souvent été confrontés les développeurs Web en JavaScript utilisant le framework Ext JS/Sencha.
L'organisation
Les questions sont organisées par thème, rendant la recherche plus facile.
Les réponses
Les réponses peuvent être complétées de liens vers d'autres réponses ou vers la documentation en ligne du framework Ext JS/Sencha.
Nouveautés et mises à jour
Lors de l'ajout ou de la modification d'une question/réponse, un indicateur est placé à côté du titre de la question. Cet indicateur reste visible pour une durée de 15 jours afin de vous permettre de voir rapidement les modifications apportées.
J'espère que cette FAQ pourra répondre à vos questions. N'hésitez pas à nous faire part de tous commentaires/ remarques/critiques.

lien : Comment participer à cette FAQ ?

Comment participer à cette FAQ ?

Auteurs : sekaijin , vermine ,

Cette FAQ est ouverte à toute collaboration. Pour éviter la multiplication des versions, il serait préférable que toute collaboration soit transmise aux administrateurs de la FAQ.
Plusieurs compétences sont actuellement recherchées pour améliorer cette FAQ.
Rédacteur : bien évidemment, toute nouvelle question/réponse est la bienvenue.
Correcteur : malgré nos efforts, des fautes d'orthographe ou de grammaire peuvent subsister. Merci de contacter les administrateurs si vous en débusquez une... Idem pour les liens erronés.

lien : Quels sont les droits de reproduction de cette FAQ ?

Quels sont les droits de reproduction de cette FAQ ?

Auteurs : sekaijin , vermine ,

Merci de contacter les auteurs pour toute copie, intégrale ou partielle de ce document, voir Comment participer à cette FAQ ?.

lien : Comment participer à cette FAQ ?

Nous tenons à remercier

Auteurs : vermine ,

L'ensemble de l'équipe des rédacteurs de www.developpez.com pour leurs remarques constructives. Nous remercions plus particulièrement sekaijin pour son implication sur le forum Ext JS et sur la création de la Faq, ainsi que ClaudeLELOUP pour sa correction rapide et indispensable.

Sommaire > Introduction

Qu'est-ce que le framework Javascript Ext-JS ?

Auteurs : sekaijin ,

Si l'on se réfère au site de Sencha. Ext-JS est le framework de choix pour développer de puissantes applications de bureau sur le web en utilisant JavaScript et les standards du web.
À son nom, est associé la petite phrase : "*Cross-Browser Rich Internet Application Framework*" qui donne bien les buts de ce framework.

Quelle est l'origine d'Ext-JS ?

Auteurs : sekaijin ,

Ext-JS est à l'origine une branche de la bibliothèque JavaScript développée par Yahoo! sous l'appellation YUI. Dès sa version 1.0, Jack Slocum, cherche à en faire un produit industrialisé. Ext-JS s'est ainsi rapidement retrouvé au rang des meilleures bibliothèques JavaScript Open Source.

Quelle licence pour Ext-JS

Auteurs : sekaijin ,

Il existe deux licences pour Ext-JS.
Il est fortement recommandé de lire les licences. Surtout si vous devez vendre vos développements.
Licence de Logiciel Commerciale :
Cette option est appropriée si vous souhaitez utiliser Ext-JS pour développer des applications commerciales dont vous restez propriétaires du code source.
Licence Open Source :
Cette licence open source est l'option appropriée si vous créez une application open source sous une licence compatible avec la GPL v3 licence GNU. Bien que la GPLv3 ait de nombreux termes, le plus important est que vous devez fournir

le code source de votre application à vos utilisateurs afin qu'ils puissent être libres de modifier votre application pour leurs propres besoins.

Qu'est-ce que Sencha ?

Auteurs : sekaijin ,

Sencha est une société commerciale créée pour fournir des produits et des prestations autour d'Ext-JS.

lien : Qu'est-ce que le framework Javascript Ext-JS ?

Qu'est-ce que Sencha Touch ?

Auteurs : sekaijin ,

Sencha Touch est un framework dérivé d'Ext-JS basé sur le HTML5 et destiné à développer des applications sur les smart phones et autres tablettes.

lien : Qu'est-ce que le framework Javascript Ext-JS ?

Qu'est-ce que Ext Core ?

Auteurs : sekaijin ,

Ext Core est le coeur du framework Ext-JS. C'est une librairie qui fournit les services de base pour le framework. Elle peut être utilisée indépendamment du framework.

Que choisir entre Ext-JS et Ext Core ?

Auteurs : sekaijin ,

Pour choisir entre les deux, il faut se pencher sur l'approche du développement.
Ext-JS est un framework destiné à développer des applications. L'approche qu'il propose est comparable à GNOME, KDE, COCOA ou WMF. L'application est construite par le code JavaScript.
Ext Core est, quant à lui, destiné à améliorer les sites web. L'approche est alors la construction d'un site web auquel on ajoute des fonctionnalités ou des composants. Cette approche est comparable à ce que propose jQuery.

lien : Qu'est-ce que Ext Core ?

Sommaire > Utiliser Ext-JS

Par quoi commencer pour faire une application Ext-JS ?

Auteurs : sekaijin ,

Ext-JS s'exécutant dans un navigateur, il est nécessaire de fournir au moins une page HTML. Vous pouvez en fournir plusieurs, mais une seule suffit.
Seule l'inclusion de la librairie et de sa feuille de style est nécessaire dans cette page.

```
<html xmlns:ext="http://extjs.com/docs">
  <head>
    <meta http-equiv="content-type" content="text/html; charset=utf-8">
    <title>Ext-JS Application</title>
    <link rel="stylesheet" href="./scripts/ext/resources/css/ext-all.css" type="text/css">
    <script type="text/javascript" src="./scripts/ext/ext-base.js"></script>
    <script type="text/javascript" src="./scripts/ext/ext-all.js"></script>
  </head>
  <body>
  </body>
</html>
```

L'inclusion de la librairie se fait au travers de deux fichiers : ext-base.js et ext-all.js.
- ext-base.js contient la définition des types et objets de base.
- ext-all.js contient la définition de tous les composants du framework.
Avec la version 4.0, le framework se charge seulement au travers du fichier ext-all.js.

```
<html>
  <head>
    <title>Account Manager</title>
    <link rel="stylesheet" type="text/css" href="ext-4.0/resources/css/ext-all.css">
    <script type="text/javascript" src="ext-4.0/ext-all.js"></script>
  </head>
  <body></body>
</html>
```

Dois-je ajouter des objets HTML particuliers ?

Auteurs : sekaijin ,

Aucune balise HTML n'est nécessaire. Toute l'interface de l'application peut être construite par le framework. À titre d'exemple voici le code type proposé par Ext-JS :

```
<html>
  <head>
    <title>Account Manager</title>
    <link rel="stylesheet" type="text/css" href="ext-4.0/resources/css/ext-all.css">
    <script type="text/javascript" src="ext-4.0/ext-all.js"></script>
    <script type="text/javascript" src="app.js"></script>
  </head>
  <body></body>
```

```
</html>
```

Puis-je ajouter des objets HTML ?

Auteurs : sekaijin ,

Oui, Ext-JS tient compte de votre code HTML. Il est conseillé d'y placer des ID afin d'en faciliter la manipulation par JavaScript.

Où dois-je placer le code de mon application ?

Auteurs : sekaijin ,

Tout le code de l'application est à placer dans le header de la page. Soit sous forme de script lié, soit de script en ligne.

Comment démarre mon application ?

Auteurs : sekaijin ,

Ext-JS utilise et complète la gestion des événements du DOM faite par JavaScript. Ext-JS fournit une méthode pour ajouter un traitement sur l'événement Ready. Cet événement est lancé lorsque l'ensemble est prêt à démarrer. Il suffit donc de placer ce code dans un script du header :

```
Ext.onReady(function() {
  // appeler les méthodes de son application.
});
```

Avec la version 4.0 vient l'introduction du pattern MVC. Si vous l'utilisez, insérez la définition de votre application :

```
Ext.application({
  ...
  launch: function() {
  }
}
```

Puis-je utiliser Ext-JS si j'ai inclus une autre bibliothèque ?

Auteurs : sekaijin ,

Oui, Ext-JS fournit un mode de compatibilité. À la place de ext-base.js, inclure le fichier correspondant à votre librairie. Ext-JS fournit les adaptateurs suivants ext-jquery-adapter.js, ext-prototype-adapter.js et ext-yui-adapter.js.

Puis-je ne charger que les composants dont j'ai besoin ?

Auteurs : sekaijin ,

Ext-all.js est un fichier compressé contenant tous les composants. Pour de petites applications, avec ses 1 051 657 octets dans sa version 4.0.0, c'est peut-être superflu. Dans le dossier pkgs d'Ext-JS se trouve un fichier par composant. Jusqu'à la version 3.x, il est nécessaire d'inclure tous les composants dont on a besoin ainsi que leurs dépendances. À partir de la version 4.0, Ext-JS propose un mécanisme de chargement dynamique.

Le framework charge automatiquement, en fonction des besoins, les différents fichiers nécessaires.

Comment définir une classe ?

Auteurs : sekaijin ,

Jusqu'à la version 3.x, Ext-JS propose la méthode Extend qui retourne une classe dérivée d'une autre classe :

```
//Ext-JS 3.x class definition
MyApp.LoginWindow = Ext.extend(Ext.Window, {
    title: 'Log in',

    initComponent: function() {
        Ext.apply(this, {
            items: [
                {
                    xtype: 'textfield',
                    name : 'username',
                    fieldLabel: 'Username'
                },
                ...
            ]
        });

        MyApp.LoginWindow.superclass.initComponent.apply(this, arguments);
    }
});
```

À partir de la version 4.0 la méthode define permet de définir une classe :

```
//Ext JS 4.x class definition
Ext.define('MyApp.LoginWindow', {
    extend: 'Ext.Window',

    title: 'Log in',

    initComponent: function() {
        Ext.apply(this, {
            items: [
                //as above
            ]
        });

        MyApp.LoginWindow.superclass.initComponent.apply(this, arguments);
    }
});
```

Merci à Ed Spencer de Sencha pour ces exemples.

Comment charger les sources nécessaires à une classe ?

Auteurs : sekaijin ,

À partir de la version 4.0 le chargement dynamique est supporté. Pour cela, lors de la définition, ajouter la liste des composants requis :

```
Ext.define('Ext.Window', {
    extend: 'Ext.Panel',
    requires: ['Ext.util.MixedCollection']
```

```
});
```

L'héritage multiple est-il supporté ?

Auteurs : sekaijin ,

Oui. Jusqu'à la version 3.x, la méthode extend permet l'héritage multiple en prenant une liste de classe en arguments :

```
//Ext-JS 3.x class definition
MyApp.LoginWindow = Ext.extend(Ext.Window, MyApp.LoginClass, {
```

La version 4.0 propose un héritage principal simple. Une classe dérive directement d'une seule autre. Mais cette version supporte un "*héritage secondaire*" de classes ou interfaces :

```
Ext.define('Sample.Musician', {
    extend: 'Sample.Person',

    mixins: {
        guitar: 'Sample.ability.CanPlayGuitar',
        compose: 'Sample.ability.CanComposeSongs',
        sing: 'Sample.ability.CanSing'
    }
})
```

Comment définir un singleton ?

Auteurs : sekaijin ,

Jusqu'à la version 3.x, Ext-JS ne propose rien de plus que ce que propose JavaScript :

```
MyApp.UserData.Connection = function(){
    ...
}();
```

À partir de la version 4.0, la méthode define qui permet de définir une classe, permet également de définir un singleton avec l'attribut singleton à true :

```
Ext.define('MyApp.UserData.Connection', {
    singleton: true
   ...
});
```

À quoi servent les namespaces ?

Auteurs : sekaijin ,

Les espaces de noms sont, dans la théorie, des ensembles de noms isolés de l'extérieur. Ainsi, on peut avoir des objets, des variables, des classes, des fonctions qui ont le même nom. Il suffit pour cela qu'ils soient définis dans des espaces de noms différents. Par exemple, Ext-JS définit une classe Window. Si pour mon application j'ai besoin d'une classe Window différente de celle de Ext-JS, je vais avoir deux classes avec le même nom, pour peu qu'elles soient définies dans des espaces de noms différents.
JavaScript ne propose pas nativement d'espace de noms. Mais il est facile d'utiliser des objets en lieu et place des espaces de noms. C'est l'approche proposée jusqu'à la version 3.x.
Ext est un objet. Tous les objets de la librairie, toutes les fonctions, les classes, sont des membres de cet objet.

À partir de la version 4, Ext-JS propose une méthode pour créer un namespace (un objet JavaScript). Mieux vaut utiliser cette méthode que de créer un objet simple comme en version 3.x. En effet, divers composants de la librairie sont capables d'utiliser un namespace ainsi déclaré :

```
Ext.namespace('Company.data');
Company.Widget = function() { ... };
Company.data.CustomStore = function(config) { ... };
```

Dois-je définir un namespace ?

Auteurs : sekaijin ,

Il est conseillé d'en définir un pour son application. Et de placer dedans tout ce dont elle a besoin.
Si vous implémentez le design pattern MVC, la première chose à faire est de définir votre application. Celle-ci doit avoir un nom. Ce nom sera un objet de type application, mais aussi un namespace dans lequel placer les éléments la concernant.
Dans la documentation sur MVC, regardez bien l'exemple. L'application porte le nom de *AM* et tous les éléments s'y rapportant y sont attachés :

```
Ext.application({
    name: 'AM',
    appFolder: 'app',
    controllers: [
        'Users'
    ],
     ...
});
Ext.define('AM.controller.Users', {
    extend: 'Ext.app.Controller',
    init: function() {
        ...
    }
});
```

lien : À quoi servent les namespaces ?

Sommaire > Modèles de conception et de développement

Ext-JS propose-t-il des modèles de conception et de développement ?

Auteurs : sekaijin ,

Ext-JS propose les modèles suivants :

- **MVC : Model View Controller ;**
- **KVC : Key Value Coding ;**
- **KVO : Key Value Observer ;**
- **Registry ;**
- **State ;**
- **Composite ;**
- **Decorator ;**
- **Delegate ;**
- **Façade ;**
- **Singleton ;**
- **Proxy.**

Suis-je obligé d'utiliser un pattern dans mon développement ?

Auteurs : sekaijin ,

Comme pour tout développement c'est mieux avec, mais ce n'est pas obligatoire.

Sommaire > Organisation du framework

Sommaire > Organisation du framework > Éléments de base > Utilitaires

Quelles sont les variables et fonctions globales définies par Ext-JS ?

Auteurs : sekaijin ,

Il n'existe qu'un seul et unique objet défini par Ext-JS : Ext.
Tous les objets, toutes les classes, toutes les méthodes dont a besoin Ext-JS sont des membres de cet objet. Cela laisse tout loisir pour choisir ses propres variables.
Pour son application, il est recommandé de définir un objet MyApp auquel on attache tout ce qui lui est nécessaire.

lien : À quoi servent les namespaces ?

Comment récupérer un objet du DOM ?

Auteurs : sekaijin ,

L'objet Ext fournit une méthode get qui permet de retrouver toute sorte d'objets. Si le paramètre fourni est l'ID d'un objet du DOM, cette méthode retourne un objet Ext.core.Element.
L'objet Ext.DomQuery permet de sélectionner des éléments du DOM en utilisant des sélecteurs CSS3 ou XPATH.

Comment insérer des éléments dans le DOM ?

Auteurs : sekaijin ,

L'objet Ext.DomHelper fournit tous les services nécessaires à la création d'éléments dans le DOM.
Il se base sur une description JSON des éléments à créer :

```
{
    id: 'my-ul',
    tag: 'ul',
    cls: 'my-list',
    // append children after creating
    children: [
        // may also specify 'cn' instead of 'children'
        {tag: 'li', id: 'item0', html: 'List Item 0'},
        {tag: 'li', id: 'item1', html: 'List Item 1'},
        {tag: 'li', id: 'item2', html: 'List Item 2'}
    ]
};
```

Pourquoi DOMHelper utilise-t-il JSON et non HTML ?

Auteurs : sekaijin ,

Cela permet de faciliter l'usage des variables. Les variables sont directement placées dans la spécification, là où il faudrait concaténer plusieurs chaines :

```
var length='45px' ;
var spec={
  tag : 'li',
  width :length
} ;
```

Comment modifier un objet du DOM ?

Auteurs : sekaijin ,

Ext.core.Element permet de manipuler un objet du DOM :

```
var el = Ext.get("my-div");
el.setWidth(100);
```

La méthode Ext.get retourne toute sorte d'objets. Si la valeur passée est l'ID d'un composant Ext-JS, c'est celui-ci qui sera retourné. Si l'ID est celui d'un élément du DOM, c'est une

instance de Ext.core.Element référençant l'élément du DOM qui est retournée. Cet objet permet de manipuler l'élément du DOM. Mais ce n'est pas l'élément du DOM.

lien : Puis-je récupérer un élément du DOM ?

Puis-je récupérer un élément du DOM ?

Auteurs : sekaijin ,

La méthode Ext.getDom est faite pour cela :

```
var el = Ext.getDom("my-div");
```

lien : Comment modifier un objet du DOM ?

Puis-je animer une modification d'un objet du DOM ?

Auteurs : sekaijin ,

Les méthodes de l'objet Ext.core.Element permettent de manipuler avec des animations un objet du DOM :

```
var el = Ext.get("my-div");
// Element animation options object
var opt = {
    duration: 1,
    easing: 'elasticIn',
    callback: this.foo,
    scope: this
};
// animation with some options set
el.setWidth(100, opt)
```

Comment connaître le navigateur ?

Auteurs : sekaijin ,

Dans la version 3.x, l'objet Ext contient un ensemble de propriétés isXXX qui valent true ou false en fonction du navigateur :

```
Ext.isOpera;
```

Dans la version 4.x, l'objet Ext contient un objet browser permettant de déterminer le navigateur :

```
Ext.browser.is.Opera;
```

Comment connaître le système ?

Auteurs : sekaijin ,

Dans la version 3.x, l'objet Ext contient un ensemble de propriétés isXXX qui valent true ou false en fonction du système :

```
Ext.isLinux;
```

Dans la version 4.x, l'objet Ext contient un objet OS permettant de déterminer le système du client :

```
Ext.os.is.Linux;
```

Comment connaître les capacités du navigateur ?

Auteurs : sekaijin ,

Dans la version 3.x, le développeur ne peut se baser que sur les informations du navigateur et de l'OS pour déterminer les capacités du navigateur. La seule information complémentaire est le moteur de rendu utilisé :

```
Ext.isWebkit ;
```

La version 4.0 introduit l'objet FeatureDetector qui permet de savoir quelles sont les capacités du client. Par exemple Geolocation, canevas, CSS3DTransforms.
Cet objet n'est, pour le moment, pas documenté. Mais son code source permet de l'utiliser :

```
Ext.features.has('SVG');
```

Qu'est-ce qu'un conteneur ?

Auteurs : sekaijin ,

Un conteneur est un objet capable d'accueillir des éléments de vue. C'est un composant générique fournissant les méthodes d'ajout et de retrait d'éléments. Il regroupe ainsi les éléments et permet une manipulation globale comme les actions d'afficher et de masquer.

Qu'est-ce qu'un panel ?

Auteurs : sekaijin ,

Un panel est un conteneur structuré permettant la gestion d'un ensemble d'éléments, et disposant également de barre d'outils, de statuts, etc.

lien : Qu'est-ce qu'un conteneur ?

Qu'est-ce qu'un layout ?

Auteurs : sekaijin ,

Un layout est un agencement permettant de définir plusieurs zones dans lesquelles il est possible de placer des éléments de vue, généralement des panneaux.

Puis-je utiliser plusieurs agencements dans la même application ?

Auteurs : sekaijin ,

Oui, un layout peut être placé dans un autre layout. C'est ce qui est utilisé par exemple pour faire une interface du style de MS Outlouk. Un layout border définit les zones nord, sud, est, ouest et centre. Dans la zone ouest est placé un layout Accordion.

Qu'est-ce qu'un modèle d'affichage ?

Auteurs : sekaijin ,

Un modèle d'affichage est un ensemble de code HTML contenant des marqueurs qui seront remplacés à l'affichage par les valeurs des variables associées.

Quand dois-je ou ne dois-je pas utiliser un modèle d'affichage ?

Auteurs : sekaijin ,

Un modèle d'affichage est assez similaire au fonctionnement des templates php du genre de SMARTY.
Cela implique une définition du code HTML, un ensemble de variables et un moteur de génération du code final.
Un affichage nécessite beaucoup d'objets en mémoire. Par contre lorsqu'on doit afficher un grand nombre de données toujours formatées de la même façon, les modèles font preuve d'efficacité.

Quel avantage a-t-on à utiliser un modèle par rapport à la concaténation de string ?

Auteurs : sekaijin ,

Une méthode courante lorsqu'on veut reproduire des affichages avec des valeurs différentes est de concaténer des chaînes contenant le code HTML avec les valeurs des variables. Le modèle d'affichage, quant à lui, va prendre la chaîne décrivant le code et la compiler pour ainsi optimiser la génération de l'affichage.

lien : Quand dois-je ou ne dois-je pas utiliser un modèle d'affichage ?

Quelles sont les différences entre template et xtemplate ?

Auteurs : sekaijin ,

Le template est un modèle simple et très rapide. Un xtemplate lui ajoute des fonctionnalités comme des fonctions mathématiques ou bien de la génération conditionnelle.
Les deux restent à disposition du développeur pour optimiser son application.

Que sont les modèles de données ?

Auteurs : sekaijin ,

Un modèle de données est une définition qui s'apparente à une définition UML. On peut voir ça comme un DataObject. Le but est de stocker et transporter des données structurées. Il ne s'agit pas ici de définir une classe avec des méthodes de manipulations mais juste un accès aux données. Contrairement à un simple objet JavaScript, un objet défini à partir d'un modèle est conforme à la définition de ce modèle.

Qu'est-ce qu'un DataStore ?

Auteurs : sekaijin ,

Un DataStore est un espace de stockage de données conforme à un modèle. Il permet la lecture et l'écriture de données. Associé à un proxy, il permet les échanges entre le client et le serveur.
Jusqu'à la version 3.x, il existe plusieurs types de DataStore en fonction de la façon dont les données sont lues. ArrayStore, JsonStore, XmlStore,...
À partir de la version 4, il n'existe que des DataStores. Le DataReader associé au DataStore est spécifique à une méthode de lecture.
Le rôle du DataStore est de faire la liaison entre un modèle, un stockage (Array, Ajax ...), un Reader et un Writer.

À quoi sert un DataStore ?

Auteurs : sekaijin ,

Il permet à tous les composants de vos applications d'avoir accès à des données de façon uniforme.
Les composants vont lire et écrire dans les DataStores quel que soit le type de stockage. Si votre application évolue ou change de protocole, mais que la définition des données ne change pas, seuls le Writer et le Reader ou le Proxy changeront.
Un bon moyen de commencer un développement est d'utiliser un Stockage local dans un Array avec un proxy LocalStorage. Les données sont alors écrites en dur dans le tableau. Une fois l'application suffisamment avancée, on peut développer la partie serveur et changer de proxy (principe de séparation des couches).

À quoi servent les proxy d'accès aux données ?

Auteurs : sekaijin ,

Le proxy est là pour faire l'interface entre la zone réelle de stockage et le DataStore. Il va ainsi relayer vers les serveurs les demandes de lecture ou d'écriture des données. Les proxy de Ext-JS supportent de nombreux protocoles.

```
var myStore = new Ext.data.Store({
    model: 'User',
    proxy: {
        type: 'ajax',
        url : '/users.json',
        reader: {
            type: 'json',
            root: 'users'
        }
    },
    autoLoad: true
```

```
});
```

Sommaire > Organisation du framework > Éléments pour le framework > Ajax

Comment faire une requête Ajax ?

Auteurs : sekaijin ,

Il faut appeler la méthode request de l'objet AJAX avec le paramètre url et la liste des paramètres sous la forme JSON :

```
Ext.Ajax.request({
    url: 'page.php',
    params: {
        id: 1
    },
    success: function(response){
        var text = response.responseText;
        // process server response here
    }
});
```

Comment récupérer les informations d'une requête Ajax ?

Auteurs : sekaijin ,

L'attribut succes permet de définir un handler traitant la réponse du serveur.
L'attribut failure permet de traiter les erreurs. Pour tout autre type d'information comme l'avancement de la requête, l'objet Ext.Ajax implémente le design pattern observable. On peut ainsi attacher des handler aux événements de son choix. Et si nécessaire, l'enrichir d'autres événements.

Comment formater une date ?

Auteurs : sekaijin ,

Ext-JS ajoute de nombreuses méthodes à l'objet Date. Les divers objets affichant des dates ont un attribut format qui définit le format d'affichage.
Lorsqu'il n'est pas renseigné, Ext-JS utilise le format par défaut.
Ce dernier est défini par la langue utilisée par le client.

Comment contraindre un nombre dans un intervalle ?

Auteurs : sekaijin ,

La méthode Contraint de Ext.Number est là pour ça. Elle ne modifie pas les valeurs des variables passées en paramètres mais retourne le nombre contraint.

Comment récupérer un nombre dans une variable de type quelconque ?

Auteurs : sekaijin ,

La méthode num de Ext.Number tente de trouver le nombre dans son premier paramètre. Si elle n'y parvient pas elle retourne son deuxième paramètre.

Sommaire > Organisation du framework > Éléments pour le framework > Utilitaires

Puis-je connaître la largeur qu'occupe une chaine à l'écran ?

Auteurs : sekaijin ,

La fonction Ext.Util.TextMerics.getHeigth retourne la largeur d'un texte en pixels.

Sommaire > Gestion des événements

Comment réagir à un événement ?

Auteurs : sekaijin ,

Ext-JS définit un ensemble de classes et d'objets servant à gérer les événements.
L'ensemble se base sur le Design Pattern Observable.
L'objet Ext.EventManager est le gestionnaire d'événements de votre application. C'est un singleton qui est défini par le framework.
Tous les événements sont encapsulés dans un objet Ext.EventObject, que ce soit des événements du système, du DOM, du framework ou de votre application.
Les objets capables d'émettre des événements dérivent (implémentent) Ext.Observable.
Lors de l'instanciation de tels objets, l'attribut listener liste les handlers que vous voulez attacher aux événements de votre objet.

```
var newEmployee = new Employee({
    name: employeeName,
    listeners: {
        quit: function() {
            // By default, "this" will be the object that fired the event
            alert(this.name + " has quit!");
        }
    }
});
```

Mais il est aussi possible d'attacher dynamiquement un handler.

```
newEmployee.on('quit', function() {
            alert(this.name + " has quit!");
        }
);
```

Comment intercepter un événement ?

Auteurs : sekaijin ,

Ext-JS attache son propre gestionnaire d'événements à chaque objet le nécessitant. Il distribue ensuite l'événement à tous les handlers qui sont attachés à l'événement. Lorsqu'on veut être certain de faire un traitement sur un événement avant que tous les traitements des handlers soient déclenchés, il suffit de définir un intercepteur avec la méthode capture.

```
Ext.util.Observable.capture(newEmployee function(event) {
        ...
    }
);
```

Comment accéder à la méthode d'un objet lors d'un événement sur un autre objet ?

Auteurs : sekaijin ,

Lorsqu'on définit un handler, la fonction fournie s'exécute dans le contexte de l'objet ayant généré l'événement. Si on passe en paramètre une méthode d'un autre objet à la méthode, lors de l'exécution, this représentera l'objet ayant émis l'événement et non celui à l'origine de la méthode (ceci est le fonctionnement de JavaScript).
Pour pallier cette difficulté, Ext-JS permet de définir un attribut scope qui sera l'objet sur lequel sera appelée la méthode :

```
Ext.define('MyApp.Job', {
    ...
    foo : function() {
        ...
    }
}
newJob = new MyApp.Job');
//On 'quit' event on newEmployee call newJob.foo()
newEmployee.on('quit', foo, newJob);
```