# Créer une grille

## Objectif :



## Mise en place :

### grid basique

Dans cet exemple on se contente de lui fournir les enregistrement (propriété ***records***). Elle se débrouille alors pour trouver les colonnes (à partir des noms des propriétés du premier enregistrement)

<!DOCTYPE html>

<html>

<head>

<title>**Grid**</title>

<meta charset=**"UTF-8"**/>

<link href=**"css/jsa.css"** rel=**"stylesheet"** media=**"all"** type=**"text/css"**>

<link href=**"css/multiDisplay.css"** rel=**"stylesheet"** media=**"all"** type=**"text/css"**>

<link href=**"css/editor2.css"** rel=**"stylesheet"** media=**"all"** type=**"text/css"**>

<link href=**"img/favicon.png"** rel=**"icon"** media=**"all"** type=**"text/css"**>

<meta name=**"viewport"** content=**"width=450, user-scalable=no"**>

<meta charset=**"utf-8"**>

<style>

</style>

</head>

<body>

<script src=**"js/jsa.js"**></script>

<script src=**"js/jsa.cls.js"**></script>

<script src=**"js/jsa.appliMobile.js"**></script>

<script src=**"js/multiDisplay.js"**></script>

<script src=**"js/grid.js"**></script>

<script src=**"js/selector.js"**></script>

<script src=**"js/editor2.js"**></script>

<script src=**"js/jsa.application.js"**></script>

<script>

</script>

<script>

jsa.appliMobile**({**

 id**:**"myOrganization"**,**

 label**:**"My organization"**,**

 objects**:[],**

 displayBody**:*function*(){**

 ***return*** **{**

 tag**:**"div"**,**

 className**:**"appliMobile\_body"**,**

 owner**:*this*,**

 isElt**:**"\_divBody"**,**

 children**:[{**

 tag**:**"grid"**,**

 records**:[**

 **{**id**:**1**,** label**:**"item 1"**,** prop1**:**"aaa1"**,** prop2**:**"bbb1"**,** prop3**:**"ccc1"**,** prop4**:**"ddd1"**},**

 **{**id**:**2**,** label**:**"item 2"**,** prop1**:**"aaa2"**,** prop2**:**"bbb2"**},**

 **{**id**:**3**,** label**:**"item 3"**,** prop1**:**"aaa3"**,** prop2**:**"bbb3"**},**

 **{**id**:**4**,** label**:**"item 4"**,** prop1**:**"aaa4"**,** prop2**:**"bbb4"**},**

 **{**id**:**5**,** label**:**"item 5"**,** prop1**:**"aaa5"**,** prop2**:**"bbb5"**},**

 **{**id**:**6**,** label**:**"item 6"**,** prop1**:**"aaa6"**,** prop2**:**"bbb6"**},**

 **{**id**:**7**,** label**:**"item 7"**,** prop1**:**"aaa7"**,** prop2**:**"bbb7"**},**

 **{**id**:**8**,** label**:**"item 8"**,** prop1**:**"aaa8"**,** prop2**:**"bbb8"**},**

 **{**id**:**9**,** label**:**"item 9"**,** prop1**:**"aaa9"**,** prop2**:**"bbb9"**},**

 **{**id**:**10**,** label**:**"item 10"**,** prop1**:**"aaa10"**,** prop2**:**"bbb10"**}**

 **]**

 **}]**

 **};**

 **}**

 **});**

appli.mobile**=*true*;**

appli.smallScreen**=*true*;**

</script>

</body>

</html>

### Avec columns

La propriété columns permet de définir :

* Les colonnes à afficher
* Leur label (optionnel)
* Leur taille (optionnel)



Voici le résultat :



### Avec fields

La propriété ***fields*** permet de définir les propriétés des différents champs des enregistrements. Le contenu de la propriété *columns* seront alors déduits de ceux-ci :



Remarquer ici

* Qu’on en profite pour définir le libellé qui se retrouvera dans les colonnes.
* On a volontairement omis la propriété *prop4*. La grille n’affichera que les colonnes correspondant à ces champs :



### A partir de la définition d’un objet

Dans l’exemple ci-dessous on se contente de fournir la définition d’un objet au moyen de la propriété ***idObj*** :



Ici *idObj* indique la définition d’objet *tag1*. La grid s’arrange alors pour y récupérer la propriété *fields* propriété *columns*.

Remarques

* On aurait pu indiquer dans l’objet la propriété ***fields*** à la place (ou en plus) de *columns*.
* Si on ne définit par *idObj* alors la grid ira chercher le tag du premier enregistrement (si présent) :



### Hauteur totale

La propriété ***height*** permet de la définir. Une barre de défilement apparaît si nécessaire :





### Détail d’un enregistrement

En cliquant sur un enregistrement il est possible d’en avoir le détail :



Pour cela il faut définir dans l’application l’objet correspondant au tag de l’enregistrement :



**Remarque**

Il est nécessaire d’inclure à minima les fichiers ***form2.js*** et ***formInput***:



### Ajout de nouveaux enregistrements

Il est possible d’ajouter de nouveaux enregistrements au moyen du bouton 



Une fenêtre apparaît alors pour saisir les champs :



**Remarque**

Par défaut la grid ajoute un enregistrement de même type que le dernier enregistrement trouvé. Mais il est possible d’indiquer explicitement quel(s) type(s) sont possibles par la propriété ***possibleChildren*** :



Dans cet exemple on indique directement les identifiants de 2 définitions d’objets

Mais on peut aussi indiquer des groupes de définitions d’objets :



Ces groupes sont définis au niveau de l’application :



possibleChildren=[{idObj:#objectDefinition} | {idGroup:#groupObjectsDefinitions}]

Si plusieurs types d’objets sont possibles :

* Un **clic court** insère un enregistrement de même type que le dernier enregistrement présent
* Un **clic long** affiche la **liste des types d’objets possibles** :



### Suppression d’enregistrements

Par un clic long sur un enregistrement la grid passe en mode sélection :



On peut alors supprimer les enregistrements au moyen du bouton 

On quitter le mode sélection par le bouton 