



ACube, Framework Ergonomique

Spécification Générale des évolutions des versions 2.9.X



Version 1.0 du 06/08/09

Etat : validé



SUIVI DES MODIFICATIONS

Version	Rédaction	Description	Vérification	Date
1.0	A Lesuffleur	Initialisation, agrégation des versions de la même branche pour avoir un seul document par version majeure		06/08/09
		<i>Document validé dans sa version xxx</i>		

LISTE DE DIFFUSION

Organisation	Nom	Info	Commentaire	Validation
		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>



SOMMAIRE

SUIVI DES MODIFICATIONS	2
LISTE DE DIFFUSION	2
SOMMAIRE	3
TABLEAUX.....	3
FIGURES.....	3
1 OBJECTIFS DU DOCUMENT	4
2 SPECIFICATION FONCTIONNELLE GENERALE.....	5
2.1 Composant arborescence	5
2.1.1 Description	5
2.1.2 Exemples	5
2.2 Composant tableau : Troncature de texte	6
2.2.1 Description	6
2.2.2 Exemple.....	7
2.3 Composant creerFormulaire.....	7
2.3.1 Description	7
2.3.2 Exemple.....	7

TABLEAUX

FIGURES

DOCUMENTS DE REFERENCE

Version	Titre



1 OBJECTIFS DU DOCUMENT

Il s'agit dans le cadre de la mise en place des évolutions au sein du Framework Ergonomique ACube de spécifier au travers de ce document le cadre fonctionnel des évolutions mises en place ainsi que leur mode de réalisation au sein de Framework.

Ce document sert donc à la fois de manuel utilisateur à destination des futurs utilisateurs du Framework, mais aussi de référence pour les futures maintenances sur le Framework.

2 SPECIFICATION FONCTIONNELLE GENERALE

Ce chapitre permet de définir dans les grandes lignes les nouvelles fonctionnalités mises en place dans le Framework Ergonomique. Il permet à un utilisateur de prendre rapidement connaissance de ces nouvelles fonctions et de leur contexte d'utilisation.

2.1 COMPOSANT ARBORESCENCE

2.1.1 DESCRIPTION

Le composant arborescence peut maintenant être utilisé pour gérer dynamiquement les nœuds.

Trois fonctions primitives ont été ajoutées pour pouvoir manipuler l'arbre :

- Créer un nœud/élément
- Modifier un nœud/élément
- Supprimer un nœud/élément

Ainsi que deux fonction JavaScript de plus haut niveau, permettant à l'application de manipuler les nœuds dans le cas d'un drag & drop par exemple :

- Ajouter un nœud/élément par rapport à un autre nœud
- Copier un nœud/élément par rapport à un autre nœud

2.1.2 EXEMPLES

```
var arbre = new ComposantArborescence(...);
```

// Récupération d'un nœud par son ID

```
var noeud = arbre.getNode("ID_DU_NOEUD");
```

// Déplacement d'un nœud dans un autre nœud

```
var noeudAdeplacer = arbre.getNode("ID_DU_NOEUD_A_DEPLACER");  
var noeudCible = arbre.getNode("ID_DU_NOEUD_CIBLE");  
arbre.addNode(noeud, noeudCible);
```

// Modification d'une propriété d'un nœud

```
var noeud = arbre.getNode("ID_DU_NOEUD");  
arbre.modifyNode(noeud, ObjectItemArborescence.GLOBALS.PROP_LIBELLE, "Nouveau libellé");
```



// Suppression d'un nœud

```
var noeud = arbre.getNode("ID_DU_NOEUD");  
arbre.deleteNode(noeud); // Equivalent à : 'arbre.deleteNode("ID_DU_NOEUD")'
```

// Création d'un nœud

```
var noeud = arbre.createNode();  
arbre.modifyNode(noeud, ObjectItemArborescence.GLOBALS.PROP_ID, "id_du_noeud");  
arbre.modifyNode(noeud, ObjectItemArborescence.GLOBALS.PROP_LIBELLE, "libelle du  
noeud");  
arbre.modifyNode(noeud, ObjectItemArborescence.GLOBALS.PROP_URL,  
"http://url_du_noeud");  
arbre.modifyNode(noeud, ObjectItemArborescence.GLOBALS.PROP_FUNCTION,  
"fonction_du_noeud");
```

// Copie d'un nœud existant

```
var copie = arbre.copyNode("ID_DU_NOEUD_A_COPIER");
```

// Ajout d'un nouveau nœud dans un nœud existant

```
var noeud = arbre.createNode();  
arbre.modifyNode(noeud, .....);  
arbre.addNode(noeud, "ID_DU_NOEUD_EXISTANT");
```

// Ne pas oublier de mettre à jour l'affichage après toute manipulation !!!!

```
arbre.ecrireBind();
```

2.2 COMPOSANT TABLEAU : TRONCATURE DE TEXTE

2.2.1 DESCRIPTION

Il est maintenant possible de définir une troncature automatique du texte, si la valeur d'une cellule d'un tableau dépasse une longueur en nombre de caractère.

Cette propriété « maxlength » est valorisée par le nœud <MAXTEXTLENGTH> du XML de configuration. Cette troncature n'est valable que pour les colonnes de type STRING. Attention : FF2 ne permet pas l'affichage des "title" sur plusieurs lignes du fait d'un bug interne (https://bugzilla.mozilla.org/show_bug.cgi?id=45375 [^]). => Troncature automatique à 80 caractères pour l'affichage de cet attribut. Problème résolu dans FF3

Si le texte d'une cellule de la colonne dont la taille du champ texte est restreinte par longueur dépasse cette valeur (en nombre de caractères), le texte est tronqué à cette valeur. Le texte dans sa version *in extenso* est affiché au survol de la souris

NB : Depuis la version 2.9.1.

2.2.2 EXEMPLE

```
<COLONNE>

    <LIBELLE>Désignation</LIBELLE>
    <TYPE>STRING</TYPE>
    <DATA>DESIGNATION</DATA>
    <LARGEUR>350</LARGEUR>
    <ALIGNEMENT>left</ALIGNEMENT>
    <TRI>CROISSANT</TRI>
    <CACHE>NON</CACHE>
    <EDITABLE>>false</EDITABLE>
    <FORMAT>testFormat () </FORMAT>
    <MAXTEXTLENGTH>20</MAXTEXTLENGTH>

</COLONNE>
```

Attention : FF2 ne permet pas l'affichage des "title" sur plusieurs lignes du fait d'un bug interne (https://bugzilla.mozilla.org/show_bug.cgi?id=45375 [^]). => Troncature automatique à 80 caractères pour l'affichage de cet attribut. Problème résolu dans FF3

2.3 COMPOSANT CREERFORMULAIRE

2.3.1 DESCRIPTION

Si on souhaite afficher un message ou une roue dentée pendant le chargement du flux de formulaire, il faut renseigner la méthode `initXMLData` en deuxième argument une chaîne de caractère désignant l'id de la « DIV » préalablement créée pour placer le message d'attente :

```
composantFormulaire.initXMLData(URL_XMLDynamiquePage, "Position_loading");
```

NB : Depuis la version 2.9.1.

2.3.2 EXEMPLE

Fonction javascript de création de formulaire avec `créerFormulaire` :

```
function construireFormulaire() {
    var testErreur = XMLStatiquePage.parseErrorXML();
    if (testErreur === 0) {
        if (XMLInfosFormulaire === null) {
            XMLInfosFormulaire = new parent.XMLObjectSauvegarde();
            XMLInfosFormulaire.sauvegardeDOM(parent.getElements(XMLStatiquePage.xmlDoc, "ELEMENTFORM") [0]);
        }
        composantFormulaire = creerFormulaire(XMLInfosFormulaire);
        composantFormulaire.initXMLData(URL_XMLDynamiquePage, "Position_loading");
        // Import et écriture des données
        composantFormulaire.importData();
    }
}
```



```
        composantFormulaire.ecrireBind( );  
    }  
}
```

Balise HTML correspondante :

```
<div id="Position_loading"></div>
```