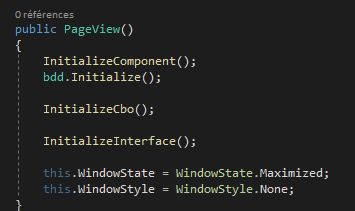
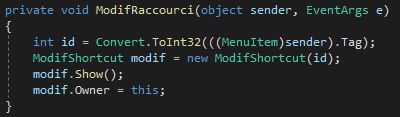
**Sommaire**

* PageView.xml.cs (p.1)
* ModifShortcut.xml.cs (p.13)
* …

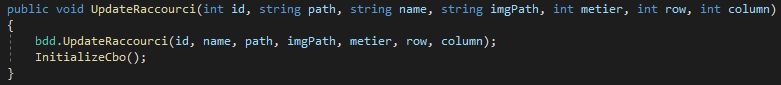
**PageView.xml.cs**



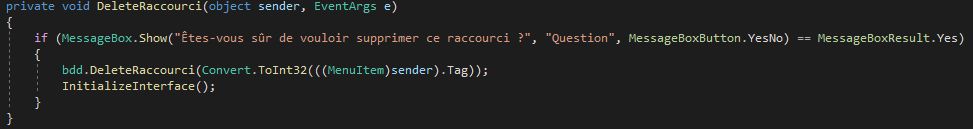
Constructeur de l'application, on appelle les fonctions qui vont servir à initialiser l'affichage ainsi que le lien avec la base de données. On demande également à la fenêtre de se mettre en taille maximal sans rebord.



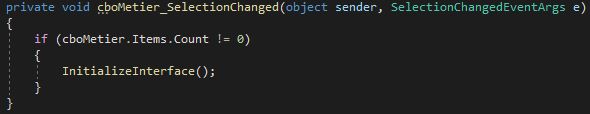
Fonction appelé par l'objet du menu contextuel lié aux boutons afin de modifier le raccourci. Elle d'appeler le constructeur de la page "ModifShortcut" avec comme paramètre l'id du raccourci avant de l'afficher. On dit aussi que la page la page principale sera le propriétaire de celle que l'on vient d'ouvrir afin que celle-ci puisse accéder à une fonction.



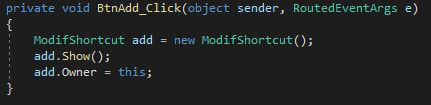
Fonction qui demande en paramètre les informations du raccourci que l'ont souhaite modifier avant d'appeler une fonction qui le modifiera dans la base de données. La fonction appelle ensuite l'initialisation de la ComboBox afin de s'assurer qu'il ne manque pas un éventuel nouveau métier créer lors de la modification. Cette fonction appellera, elle, l'initialisation de l'interface par effet de ricochet avec lorsque l'objet sélectionné dans la ComboBox change.



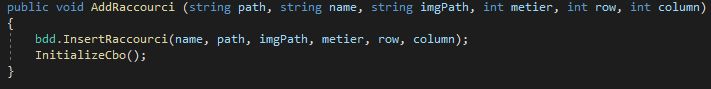
Affiche une MessageBox pour demander confirmation à l'utilisateur avant d'appeler une fonction qui supprimera le raccourci dans la base de données qui a comme paramètre l'id de celui-ci. L'interface est également réinitialisée afin de supprimer le raccourci de l'interface.



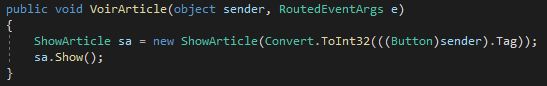
Réinitialise l'interface lorsque l'objet sélectionné dans la ComboBox change afin d'avoir les bons raccourcis affichés avec une condition pour ne pas créer de boucle infinie.



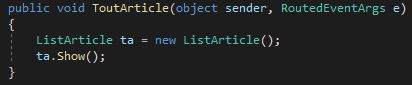
Fonction appelé lorsque l'on clique sur le bouton "Ajouter un raccourci" qui va appeler le constructeur de ModifShortcut qui va demander de créer un raccourci et non pas d'en modifier un existant avant d'afficher la page. On dit aussi que la page la page principale sera le propriétaire de celle que l'on vient d'ouvrir afin que celle-ci puisse accéder à une fonction.



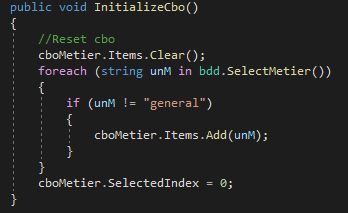
Fonction qui demande en paramètre les informations sur le raccourci que l'on souhaite créer avant d'appeler une autre fonction qui va le créer dans la base de données. La fonction appelle ensuite l'initialisation de la ComboBox afin de s'assurer qu'il ne manque pas un éventuel nouveau métier créer lors de la modification. Cette fonction appellera, elle, l'initialisation de l'interface par effet de ricochet avec lorsque l'objet sélectionné dans la ComboBox change.



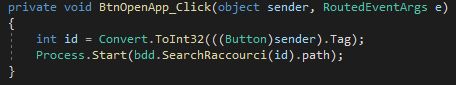
Fonction qui appelle le constructeur de ShowArticle avec l'id de l'article que l'on souhaite voir afin d'afficher cette fenêtre.



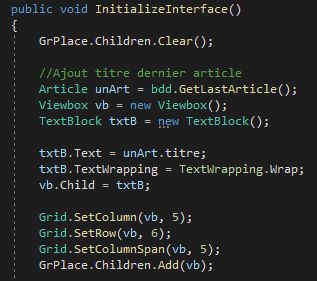
Fonction affiche la fenêtre ListArticle



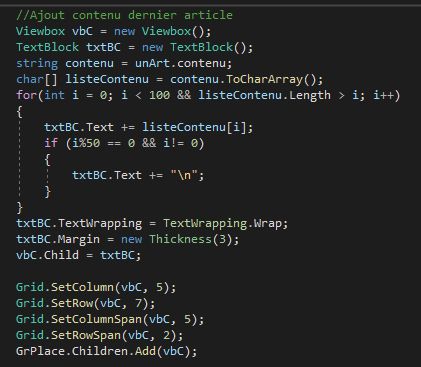
Fonction qui permet l'initialisation de la ComboBox, celle-ci est vidé avant de parcourir la liste des métiers récupérer depuis la base de données à l'exception de "général". On met également l'index par défaut sur 0 pour ne pas créer de problème.



Fonction qui récupère le tag du bouton qu'on a défini comme étant l'id du raccourci auquel il correspond, on demande ensuite à l'application ou au fichier de s'ouvrir avec comme paramètre le chemin que l'on récupère depuis la base de données grâce à l'id du raccourci.



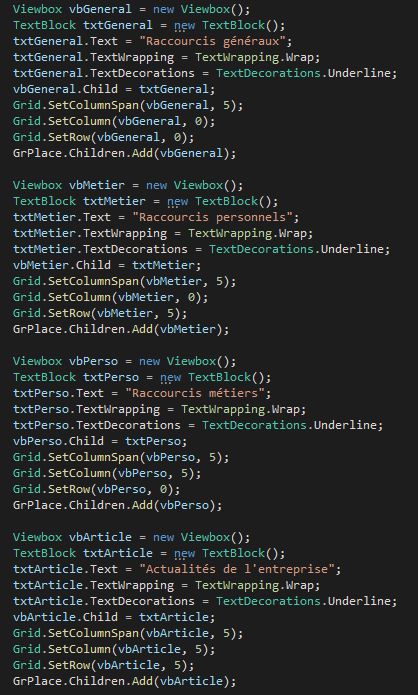
Cette partie de la fonction permet de vider le Grid, on récupère ensuite le dernier article afin d'afficher son titre dans un TextBlock se trouvant lui-même dans une ViewBox afin que celui-ci prenne toute la place disponible dans la cellule du Grid qui lui est dédié. On utilise ensuite les fonctions de Grid afin de placer le titre à l'endroit que l'on veut, ici dans la colonne indexé 5 et la ligne indexé 6 avec également un Span de 5 pour prendre toute la longueur du cadre.



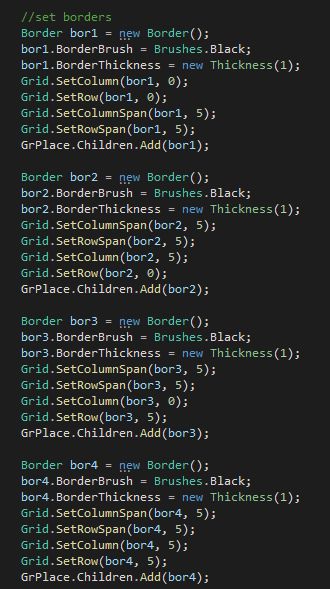
On fait sensiblement la même chose que pour le titre à l'exception qu'on va créer une liste de caractère afin d'y stocker le contenu de l'article. On va ensuite afficher les lettres une à une jusqu'à en afficher 100 pour ne pas se retrouver avec tout le contenu afficher dans une toute petite case. On fait également un saut à la ligne à 50 caractères et on ajoute un Margin afin de rendre l'affichage plus joli.



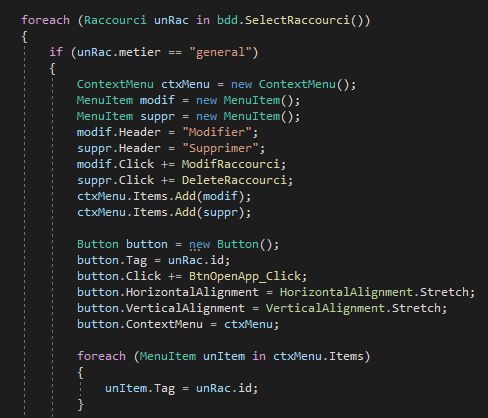
On crée des boutons qui serviront à appeler les fonctions "VoirArticle" et "ToutArticle". On leur assigne des paramètres tel que l'idée du dernier article pour le bouton permettant de voir le dernier article, leur position dans le Grid etc. Leur contenu est dans une ViewBox pour les mêmes raisons que plus haut.



On crée les textes qui serviront de titre à chacun des 4 cadres.



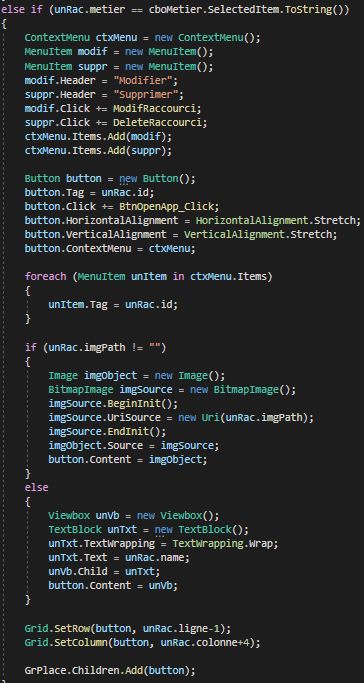
On crée des border aux 4 cadres avec leur épaisseur, leur position, leur Span etc..



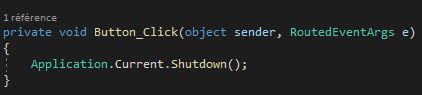
On utilise un Foreach qui va parcourir les raccourcis présents dans la base de données, si c'est un raccourci général alors on lui crée un bouton, un menu contextuel etc, on assigne comme tag au bouton et aux MenuItem l'id du raccourci afin de pouvoir les récupérer dans leur fonction OnClick.



On utilise un If pour vérifier si le raccourcis possède une icône, si c'est le cas alors on créer une Image et une BitMapImage qui serviront de contenu au bouton, si ce n'est pas le cas alors on donne le nom du raccourci comme contenu au bouton, ensuite on donne l'emplacement du bouton et on l'instancie dans la Grid.

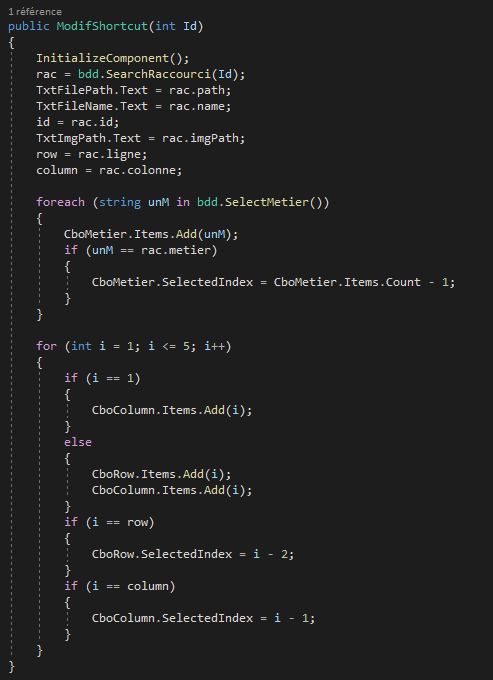
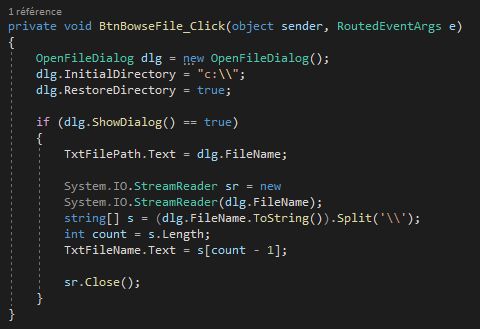
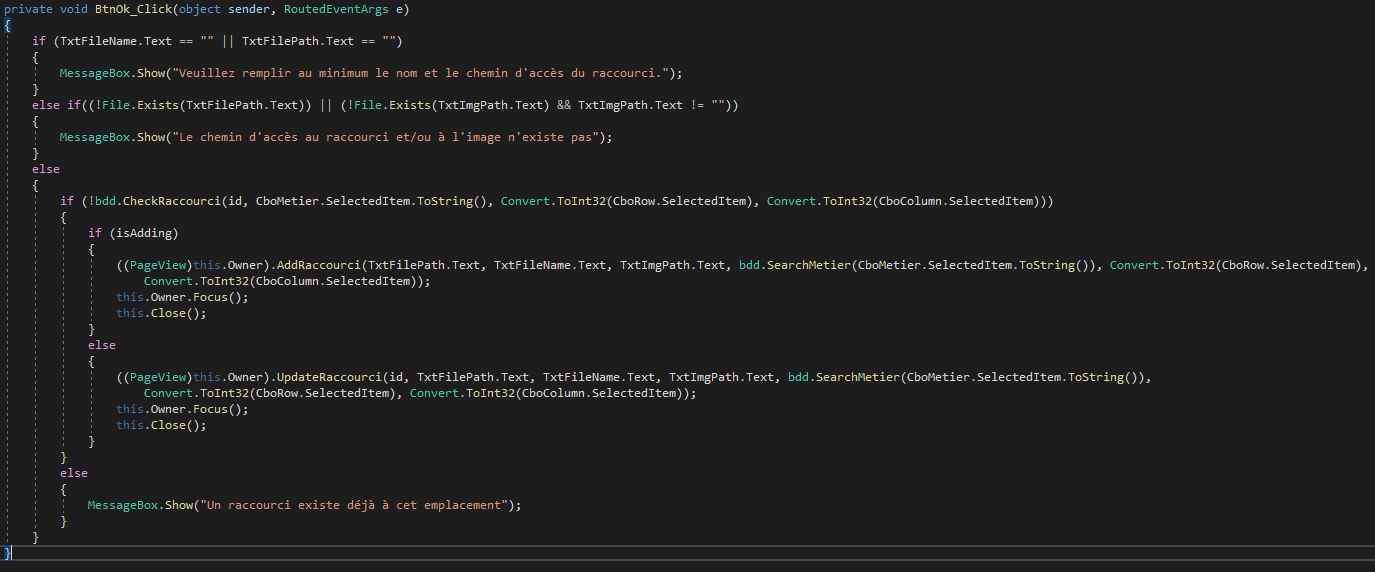
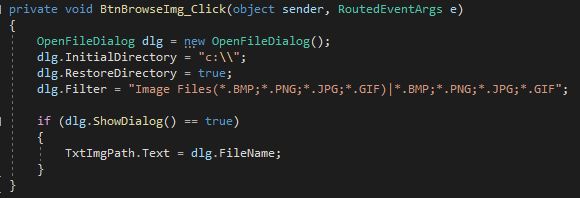
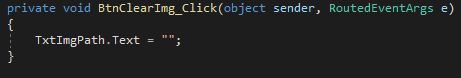
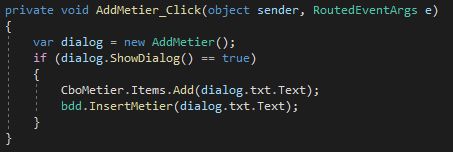
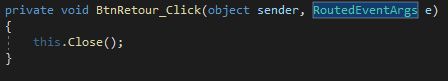
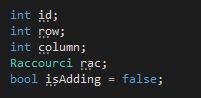


Si ce n'est pas un raccourci faisant parti de la catégorie "général" alors on vérifie qu'il fait partie du métier que l'on a sélectionné dans la ComboBox, si c'est le cas alors on fait la même chose que si c'était un "général" mais on le place 5 colonne plus loin, dans le cadre dédier aux raccourcis métier.

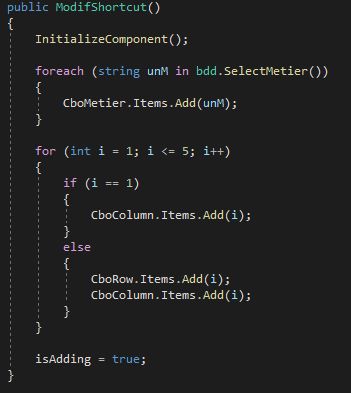


Fonction appelé par le bouton quitter qui ferme simplement l'application.

**ModifShortcut.xml.cs**

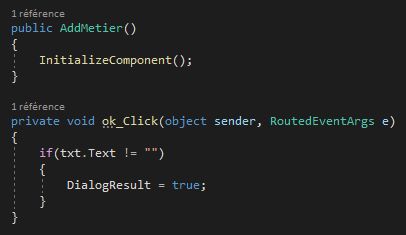


Constructeur qui demande l'id du raccourci et qui va dire au logicielle que l'on veut modifier

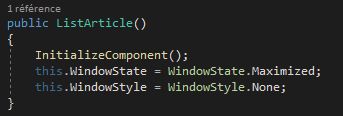
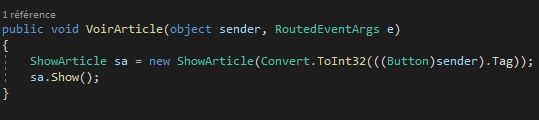
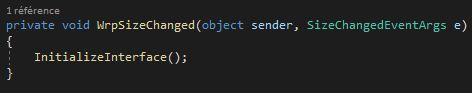
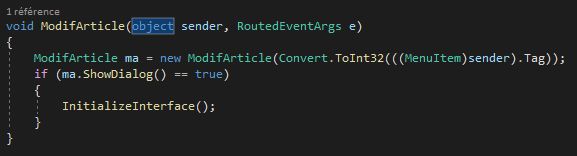
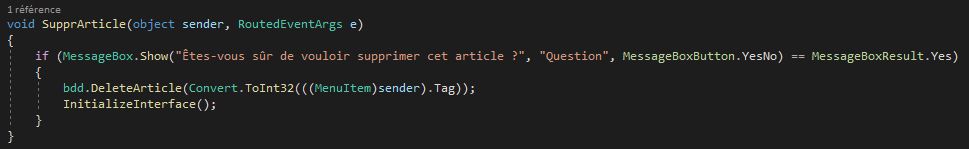
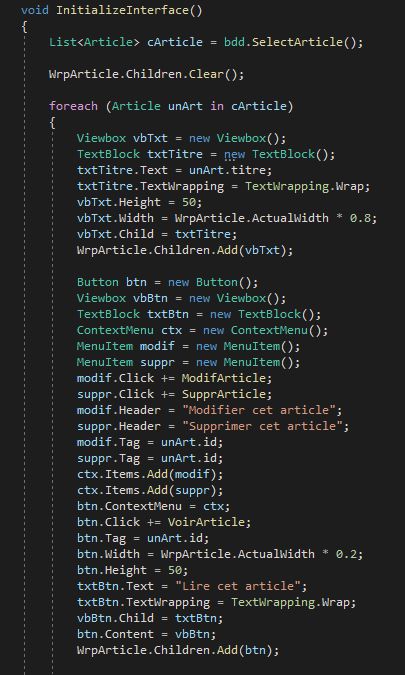
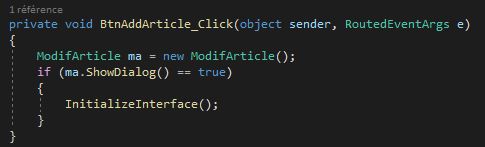
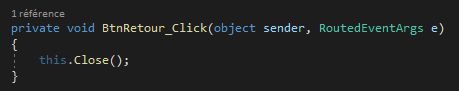


Constructeur qui dit au logiciel que l'on veut ajouter un raccourci et non le supprimer

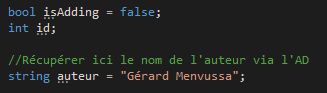
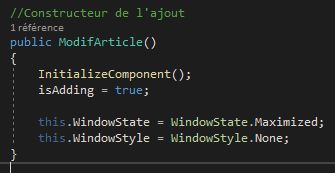
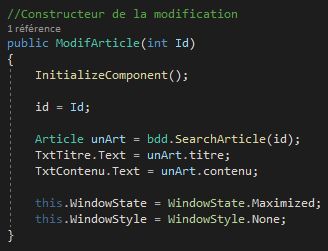
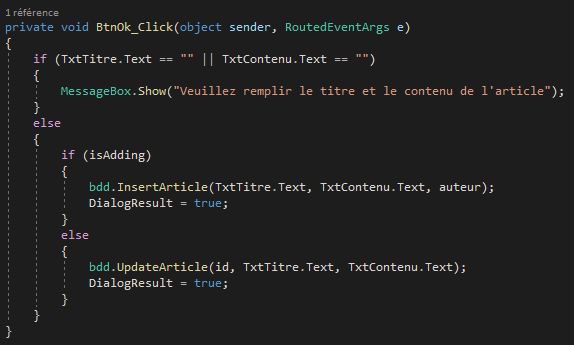
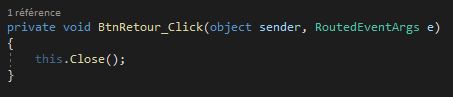
**AddMetier.xml.cs**



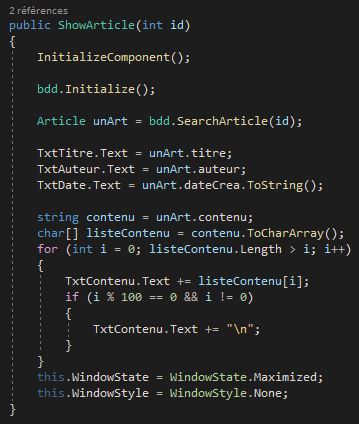
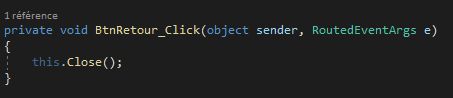
**ListArticle.xml.cs**



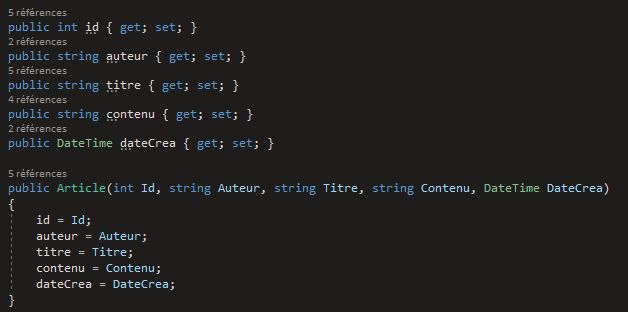
**ModifArticle.xml.cs**



**ShowArticle.xml.cs**



**Article.cs**



**bdd.cs**

