Aide au Développement sous Visual Studio

Fait le 23/02/2023 MC Batut

# Convention des noms d’objets utilisés avec VS pour Geedoc

|  |  |  |
| --- | --- | --- |
|  | Début | Exemples |
| Forme | frm | frmAnimal, frmListerAnimal |
| splitContainer | splcnt | splcntLister, splcntInventaire |
| panel | pan | panDepart, PanFact |
| groupBox | gbx | gbxMere, gbxPere |
| tabControl | tab | tabLister, TabInventaire |
| page dans les tabControl | pg | pgSelection, pgAfficher, pgChoixColonnes,pgInventaire |
| button | bt | btSaisie\_Valider, btLister\_Valider |
| textBox | txt | txtSaisie\_Libelle, txtMAJ\_CommentNais |
| label | lbl | lblSaisie\_DateNais |
| comboBox | cb | cbSaisie\_Sexe |
| radioButton | rdb | rdbLister\_Repro, chkLister\_NonRepro |
| checkBox | chk | chkSaisie\_CouleurNais |
| ListBox | lst |  |
| DateTimePicker | dtp | dtpSaisie\_DateNais |
|  |  |  |
| DataGridView | dgrv | dgrvLister, dgrvFemelle, dgrvSaillie |
| ContextMenuStrip | cmOp | cmOpLister, cmOpFemelle |
| menu | menu | menuLister\_Supprimer, menuLister\_VoirAnimal |
|  |  |  |

Objets Non Visuels : en bas de la forme

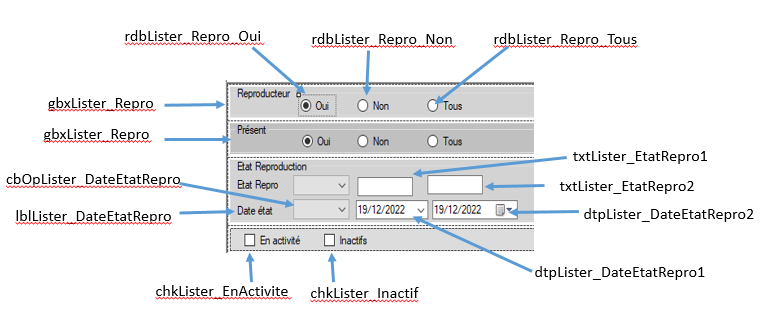
Menu, ContextMenuStrip, ToolTip, SaveFileDialog, OpenFileDialog



SaveFileDialog : pour écrire dans un fichier

OpenFileDialog : pour aller lire dans un fichier

## Exemple dans frmListerAnimaux

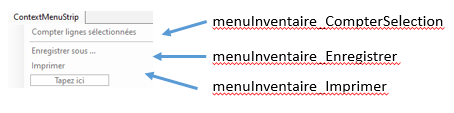


## Lien entre DataGridView et ContexMenuStrip

Un DataGridView : dgrvInventaire

Un ContextMenuStrip : cmOpInventaire





Le lien se fait par la propriété ContextMenuStrip :



# Propriétés des objets

ComboBox : mettre la propriété DropDownStyle à DropDownList pour n’accepter que les valeurs proposées

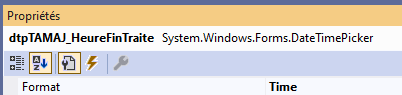


DateTimePicker : mettre format propriété Format Short pour avoir la date format DD/MM/YYYY



Mettre propriété Format=Time pour avoir Heure :Minute :Seconde





DataGridView : mettre la plupart du temps propriété SelectionMode = FulRowSelect

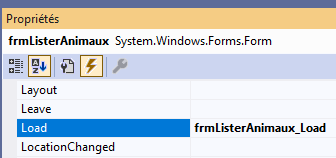


textBox : mettre un MaxLength et mettre en Jaune si à Saisir ou Jaune clair Non à Saisir

# Evènements liés aux objets

En général mais cela n’est pas toujours le cas :

**Pour un objet Form :** Load



**Dans le load** : appel à une fonction qui dit les droits Insert, Update, Delete sur la table générale

Initialisations des tailles maximales des TextBox : les dimensions sont dans clsDimGeedoc

txtLister\_BoucleVisuelle1.MaxLength = clsDimGenGeedoc.dimBoucleVisuelle;

txtLister\_BoucleVisuelle2.MaxLength = clsDimGenGeedoc.dimBoucleVisuelle;

txtLister\_ElvNais1.MaxLength = clsDimGenGeedoc.dimElvNe;

txtLister\_ElvNais2.MaxLength = clsDimGenGeedoc.dimElvNe;

Initialisations des comboBox,

Initialisations des champs pour l’inventaire, .etc..

**Dans le Closing :**



private void frmListerAnimaux\_FormClosing(object sender, FormClosingEventArgs e)

{

ds\_gen.Dispose();

ora\_gen.bdClose();

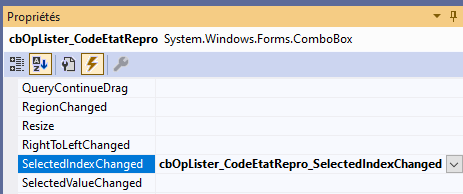
ora\_gen.bdDispose();

// Vider le Garbage Collector

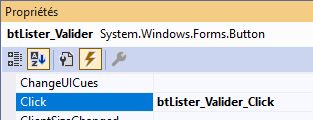
GC.Collect();

}

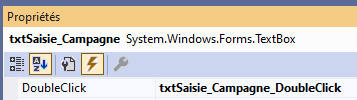
**Pour un objet ComboBox :** selectIndexChanged



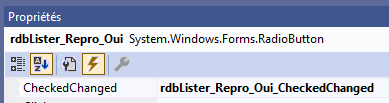
**Pour un objet boutton :** click



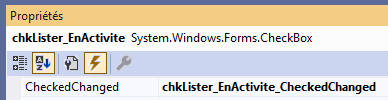
**Pour un objet textBox :** DoubleClick ou un Click



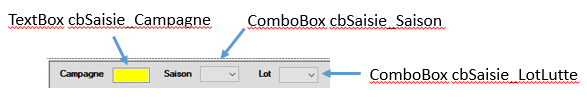
**Pour un objet radioButton :** ChechedChanged



**Pour un objet checkBox :** CheckedChanged



**Exemple : pour des saisies liées à la Reproduction**, il faudra Campagne/Saison et parfois le lot de lutte



Double clic sur la Campagne : on construire le ComboBox cbSaisie\_Saison

private void txtSaisie\_Campagne\_DoubleClick(object sender, EventArgs e)

{

clsGestionGeedoc.TableCampagne\_ConstruireComboBox\_SaisonPG2(ora\_gen, cbSaisie\_Saison, txtSaisie\_Campagne.Text);

}

Et quand l’utilisateur va choisir la Saison, on va construire le ComboBox lié aux lots de lutte intra campagne /saison

private void cbSaisie\_Saison\_SelectedIndexChanged(object sender, EventArgs e)

{

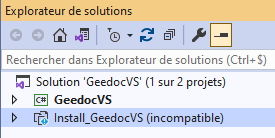
clsGestionGeedoc.TableLotLutte\_ConstruireComboBox\_LotLuttePG2(ora\_gen, cbSaisie\_LotLutte, txtSaisie\_Campagne.Text, cbSaisie\_Saison.Text, true);

}

# Organisation pour Geedoc

## L’application Geedoc : Ici GeedocVS

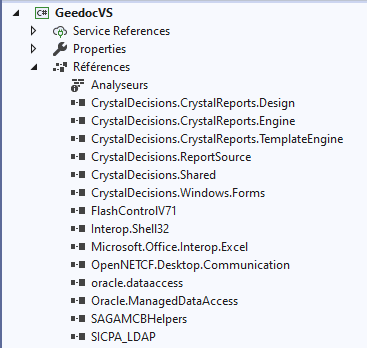
Il y a 1 solution et 2 projets mais ici mon projet pour déployer a un Pb)



1 projet GeedocVS avec tous les dossiers et dans chaque dossier des formes ou des classes

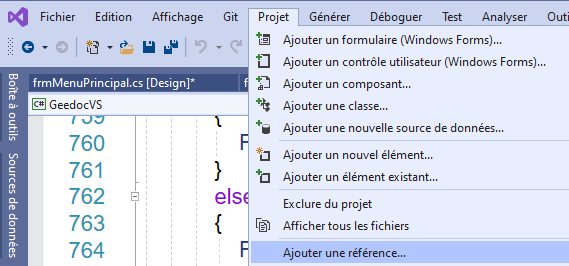
1 projet pour le déploiement (Pour faire le SetUp.exe : ici il y a un Pb)

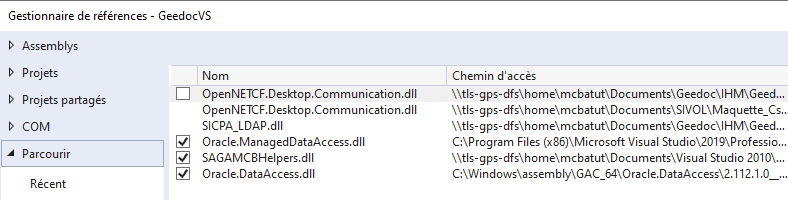
## Ajout de Dll ou de Référence



Il faut ajouter les Dll SAGAMCBHelpers.dll, SICPA\_LDAP.dll, OpenNETCF.Desktop.Communication

Pour Ajouter une DLL.





Cliquer sur « **Parcourir** » puis aller chercher la Dll.

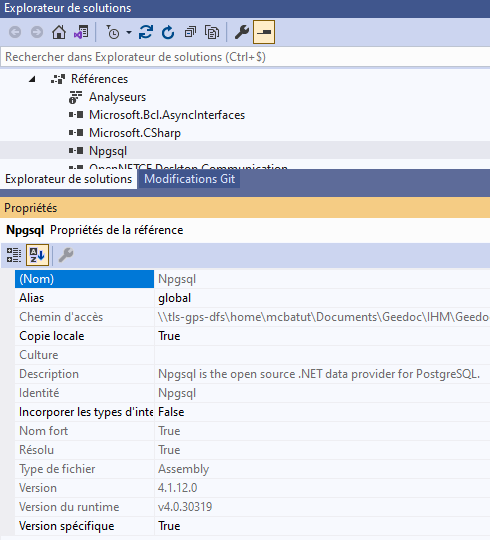
**Crystal Report** : on verra plus tard

**Pour ajouter l’accès aux WebServices de Sicpa Sanitaire** : On verra plus tard

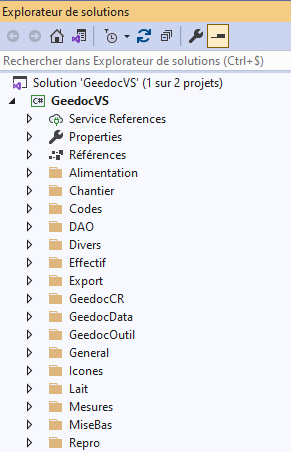
### Pour accéder à la Base de Données

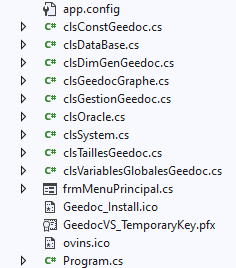
Version Oracle : oracle.dataaccess ()

Version PostgreSql :



### Les répertoires de travail





clsDataBase.cls : fonctions génériques pour atteindre la base de données de n’importe quelle nature (Oracle, PostgreSql ou MyMsql)

clsOracle.cs : fonctions spécifiques pour atteindre une base Oracle

clsConstGeedoc.cs : contient toutes les constantes

public const int MALE = 1;

public const int FEMELLE = 2;

public const int NON\_SEXE = 3;

clsDimGenGeedoc.cs : dimension générale utilisé pour les TextBox

static public int dimBoucleVisuelle = 5;

static public int dimBoucleRFID = 15; // la vrai boucle RFID

clsTaillesGeedoc.cls : taille des colonnes dans les DataGridView

static public int tailleCampagne = 40;

static public int tailleCampagneSaison = 60;

static public int tailleCauseReforme = 20;

clsVariablesGlobalesGeedoc.cs : variables globales

static public string myIdUser = "";

static public string myEspece = "";

static public string myDomaine = "";

static public string myProd = "";

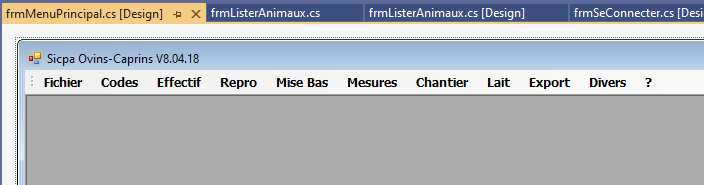
clsGeedocGraphe.cs : pour faire des graphes

clsGestionGeedoc.cls : Fonctions divers liées aux tables (création de liste d’animaux, chercher dans une liste, . etc..)

clsSystem.cs : fonctions données par Hervé Lagant pour tester si 32bist ou 64bits

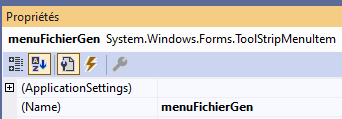
## Les Menus

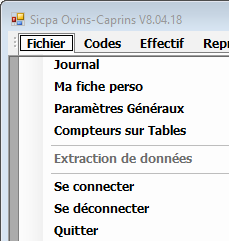
**Forme Principale : frmMenuPrincipal**

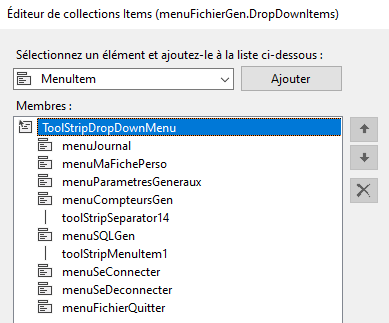




menuGen : Menu



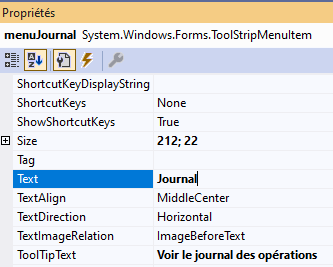




Timer :

ttInfoBulle : pour afficher des infos quand l’utilisateur met la souris sur un élément

Propriété ToolTipText



Sur chaque menu : Evènement click

private void menuJournal\_Click(object sender, EventArgs e)

{

frmJournal Fille = new frmJournal();

if (FenetreDejaActive(Fille))

{

Fille.Close();

}

else

{

Fille.MdiParent = this;

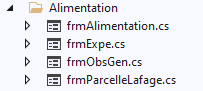
Fille.Show();

}

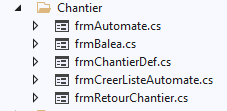
}

## Les Répertoires : formes classées par catégorie (plus ou moins)

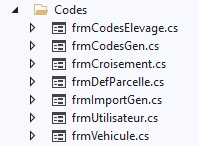
**Répertoire Alimentation**



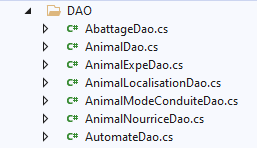
**Répertoire Chantier**



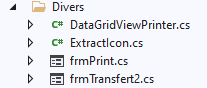
**Répertoire Codes**



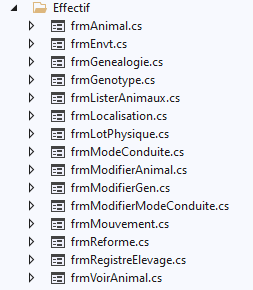
**Répertoire DAO : Data Access Objet (Insert/Update/Delete)**

****

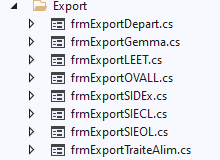
**Répertoire Divers**



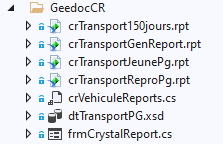
**Répertoire Effectif**



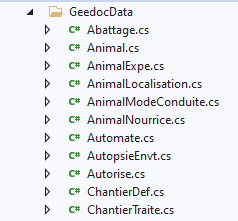
**Répertoire Export**



**Répertoire Crystal Report**



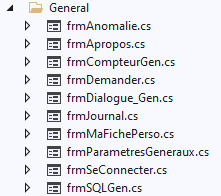
**Répertoire GeedocData :** Toutes les classes d’objets liées aux tables de données

****

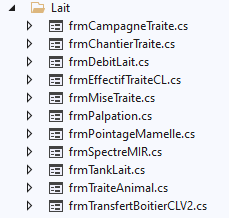
**Répertoire GeedocOutil**



**Répertoire General**

****

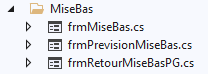
**Répertoire Lait**



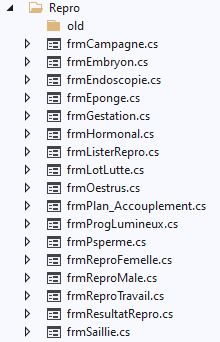
**Répertoire Mesures**



**Répertoire MiseBas**



**Répertoire Repro**



Remarque : mettre frmProgLumineux dans le menu Mesures

# Convention des couleurs dans Geedoc



En Jaune : A saisir

En jaune clair : info qui s’affiche, on ne peut pas saisir



En vert : les totaux = nombre de lignes affichées

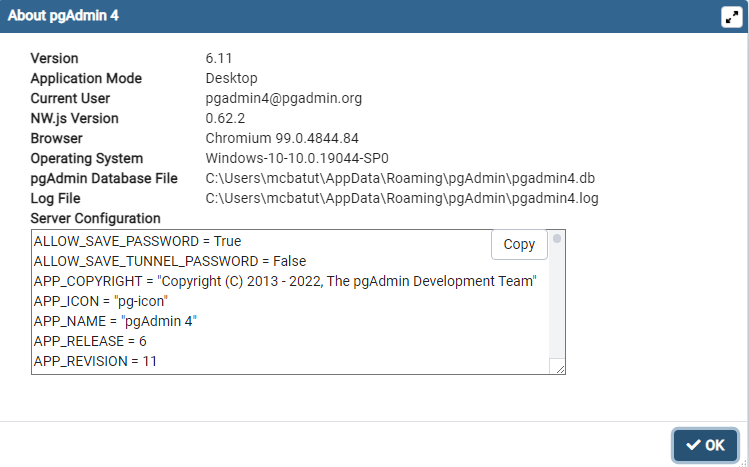
En Orange : dépend ici on peut saisir l’IPG puis double cliquer pour aller se positionner directement sur l’animal

Les couleurs sont dans clsSAGAMCBCouleur qui est dans la dll clsSAGAMCBHerlper.dll

txtSaisie\_Campagne.BackColor = clsSAGAMCBCouleur.coulInfoNePasSaisir;

cbSaisie\_Saison.BackColor = clsSAGAMCBCouleur.coulInfoNePasSaisir;

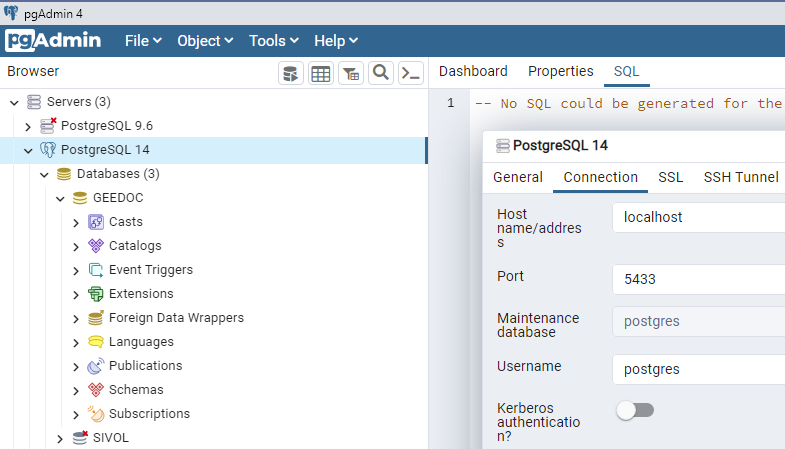
# Lien avec la base de Données



Avec les autres membres du cati, On a choisi la version 14 de PostGreSql.



Ici accès à ma base GEEDOC sur mon PC host=localhost, port=5433



Dans clsVariablesGlobales, il faut mettre iDB\_USED=3 pour PostgreSql

static class clsVariablesGlobalesGeedoc

{

// Pour se connecter à la base de données au CTIG

// OK ca marche pour PostGreSQL mon PC V2

public const string dbDataSource = "host=localhost; port=5433; database=GEEDOC"; public const string dbNameDataBase = "geedoc";

//static public int iDB\_USED = 1; // Oracle

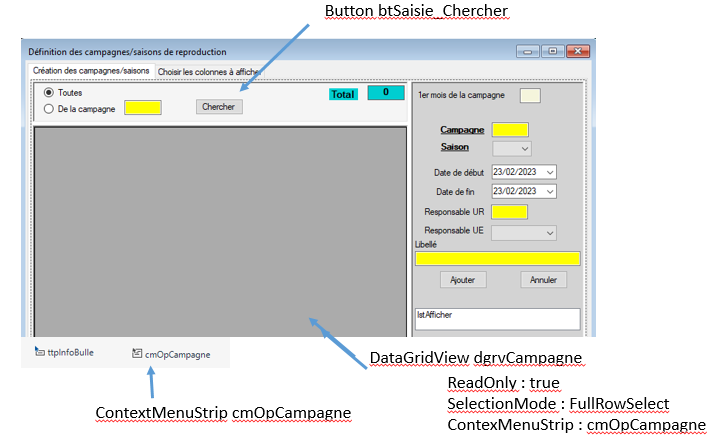
// public int iDB\_USED = 2; // MySQL

static public int iDB\_USED = 3; // PostGres

…

}

## Afficher des données dans un DataGridView



**DataGridView dgrvCampagne**

ReadOnly=true : que en lecture, on ne peut pas modifier les données dans le DataGridView

SelectionMode = FullRowSelect : toute la ligne est en bleu quand l’utilisateur se positionne dessus

ContextMenustrip : lien avec le ContextMenu

**btsaisie\_Chercher**  Si on clique sur le boutton : va lire les données et les afficher dans le DataFridView dgrvCampagne

private void btSaisie\_Chercher\_Click(object sender, EventArgs e)

{

Campagne\_RechargerDonneesV2();

}

Faire le SQL : SELECT … FROM LEFT JOIN .. WHERE

Déclaration dans la forme en haut

private DataSet ds\_gen;

private clsDataBase ora\_gen = null;

Pour cet exemple sur les Campagnes, on va travailler sur la classe DefCampagne

namespace GeedocData

{

class DefCampagne

{

private int camprepro\_id;

private int troupeau\_id; // est-ce utile ?

private string campagne;

private string saison;

private string libelleCampagne;

private DateTime dateDebutCampagne;

private DateTime dateFinCampagne;

private string responsableUr;

private string responsableUe;

/// <summary>

/// Permet de recharger/afficher les données de campagne

/// </summary>

private void Campagne\_RechargerDonnees()

{

string sql = "";

sql = "SELECT CAMPAGNE,SAISON,D\_DEBUT,D\_FIN,ID\_RESP\_UR,ID\_RESP\_UE,LIB\_CAMPAGNE,";

sql += "D\_IA\_MIN,D\_IA\_MAX,D\_LN\_MIN,D\_LN\_MAX,D\_DG\_MIN,D\_DG\_MAX,D\_MB\_MIN,D\_MB\_MAX";

sql +=",BD\_SAISIE,d\_creation,UT.id\_user as id\_createur";

sql += " FROM "+ dbNameDataBase + ".campagne CA";

sql +=" LEFT JOIN "+dbNameDataBase+".utilisateur UT ON CA.createur\_id=UT.user\_id";

sql += " WHERE troupeau\_id=" + clsVariablesGlobalesGeedoc.myTroupeauId;

if (rdbSaisie\_Campagne.Checked)

sql += " AND campagne='" + txtSaisie\_CampagneChercher.Text + "'";

sql += " ORDER BY campagne DESC,saison DESC";

string nomTable = "CAMPAGNE";

// Mettre le curseur en mode Wait

Cursor.Current = Cursors.WaitCursor;

// Lancer le SQL

if (ora\_gen.SQLResult\_In\_Dataset(ds\_gen, sql, nomTable))

{

// Lien entre la source de données nomTable et le DataGridView

dgrvCampagne.DataSource = ds\_gen.Tables[nomTable];

// Mettre le DataGridView en lecture seulement

dgrvCampagne.ReadOnly = true;

// Afficher le titre de chaque colonne

Campagne\_Afficher\_TitreColonnes(dgrvCampagne);

// Afficher le total = nb d’enregistrements affichés

txtSaisie\_TotalNb.Text = dgrvCampagne.RowCount.ToString();

txtSaisie\_TotalNb.BackColor = clsSAGAMCBCouleur.coulTotal;

if (dgrvCampagne.RowCount == 0)

{

// Ne pas afficher le ContextMenuStrip

cmOpCampagne.Enabled = false;

}

else

{

// Afficher le ContextMenuStrip

cmOpCampagne.Enabled = true;

menuCampagne\_Compter.Enabled = true;

menuCampagne\_Enregistrer.Enabled = true;

menuCampagne\_Imprimer.Enabled = true;

// Il faut avoir le droit Update

menuCampagne\_Modifier.Enabled = myDroit\_Update ;

// Il faut avoir le droit Delete

menuCampagne\_Supprimer.Enabled = myDroit\_Delete;

menuCampagne\_CampagneSaisonRepro.Enabled = true;

menuCampagne\_CampagneSaisonJeune.Enabled = true;

}

// Avoir les droits Insert pour pouvoir Ajouter

btSaisie\_Valider.Enabled = myDroit\_Insert;

}

// revenir au Curseur par Défaut

Cursor.Current = Cursors.Default;

}

ora\_gen.SQLResult\_In\_Dataset(ds\_gen, sql, nomTable) : fait le SQL et met les données dans le DataSet ds\_gen pour la collection nomTable

Pour que les données s’affichent dans le DataGridView :

dgrvCampagne.DataSource = ds\_gen.Tables[nomTable]

Campagne\_Afficher\_TitreColonnes(dgrvCampagne) : Va afficher le nom des colonnes et mettre les couleurs

/// <summary>

/// Permet d'afficher le titre des colonnes dans DataGridView à notre convention

/// </summary>

/// <param name="dgrv">DataGridView concerné</param>

private void Campagne\_Afficher\_TitreColonnes(DataGridView dgrv)

{

// Modifier le titre des colonnes

dgrv.Columns["CAMPAGNE"].HeaderText = "Campagne";

dgrv.Columns["SAISON"].HeaderText = "Saison";

dgrv.Columns["D\_DEBUT"].HeaderText = "Date de début";

dgrv.Columns["D\_FIN"].HeaderText = "Date de fin";

dgrv.Columns["ID\_RESP\_UR"].HeaderText = "Resp scientifique";

dgrv.Columns["ID\_RESP\_UE"].HeaderText = "Resp technique";

dgrv.Columns["LIB\_CAMPAGNE"].HeaderText = "Libellé";

dgrv.Columns["D\_IA\_MIN"].HeaderText = "1ère IA";

dgrv.Columns["D\_IA\_MAX"].HeaderText = "Dernière IA";

dgrv.Columns["D\_LN\_MIN"].HeaderText = "1ère LN";

dgrv.Columns["D\_LN\_MAX"].HeaderText = "Dernière LN";

dgrv.Columns["D\_DG\_MIN"].HeaderText = "1er DG";

dgrv.Columns["D\_DG\_MAX"].HeaderText = "Dernier DG";

dgrv.Columns["D\_MB\_MIN"].HeaderText = "1ère MB";

dgrv.Columns["D\_MB\_MAX"].HeaderText = "Dernière MB";

dgrv.Columns["BD\_SAISIE"].HeaderText = "BD Saisie";

dgrv.Columns["D\_CREATION"].HeaderText = "Ajouté le";

dgrv.Columns["ID\_CREATEUR"].HeaderText = "Par";

// Initialiser les couleurs

dgrv.ColumnHeadersDefaultCellStyle.BackColor = clsSAGAMCBCouleur.coulColEnTete;

dgrv.RowHeadersDefaultCellStyle.BackColor = clsSAGAMCBCouleur.coulLigneEnTete;

for (int i = 0; i < dgrv.ColumnCount; i++)

dgrv.Columns[i].DefaultCellStyle.BackColor=clsSAGAMCBCouleur.coulLigneEnTete;

dgrv.Columns["CAMPAGNE"].DefaultCellStyle.BackColor=clsSAGAMCBCouleur.coulColCleNum;

dgrv.Columns["SAISON"].DefaultCellStyle.BackColor = clsSAGAMCBCouleur.coulColCleNum;

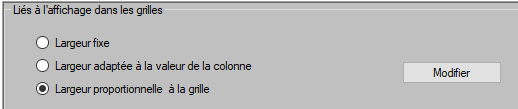
// Dimensionner les colonnes de la grille dgrv passée en paramètre

Campagne\_Redimensionner\_dgrvCampagne(dgrv);

}

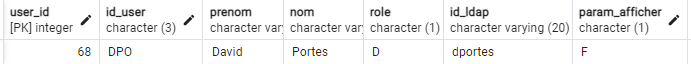
Plusieurs mode d’affichage.

Ce paramètre peut être changé dans la fiche « ***Ma Fiche Perso*** »



Mise à jour du champ param\_afficher dans la table Utilisateur

F=Largeur Fixe, V=Largeur proportionnelle à la grille, Variable C=Largeur adapté à la Colonne



Cette info est lue au moment de la connexion et est stockée dans la variable globale clsVariablesGlobalesGeedoc.myParamAfficher

/// <summary>

/// Permet de redimensionner les colonnes du DataGridView passé en paramètre

/// </summary>

/// <param name="dgrv">DataGridView à redimensionner</param>

private void Campagne\_Redimensionner\_dgrvCampagne(DataGridView dgrv)

{

if ((dgrv.Width > 0) && (dgrv.Rows.Count > 0))

{

dgrv.RowHeadersWidth = clsSAGAMCBCouleur.tailleLigneEnTete;

if (clsVariablesGlobalesGeedoc.myParamAfficher == "F")

{

#region Taille Fixe

dgrv.Columns["CAMPAGNE"].Width = clsTaillesGeedoc.tailleCampagne;

dgrv.Columns["SAISON"].Width = clsTaillesGeedoc.tailleSaison;

dgrv.Columns["D\_DEBUT"].Width = clsTaillesGeedoc.tailleDate;

dgrv.Columns["D\_FIN"].Width = clsTaillesGeedoc.tailleDate;

dgrv.Columns["ID\_RESP\_UR"].Width = clsTaillesGeedoc.tailleIdUser;

dgrv.Columns["ID\_RESP\_UE"].Width = clsTaillesGeedoc.tailleIdUser;

//dgrv.Columns["LIB\_CAMPAGNE"].Width = clsTaillesGeedoc.tailleLibelle;

dgrv.Columns["LIB\_CAMPAGNE"].AutoSizeMode = DataGridViewAutoSizeColumnMode.AllCellsExceptHeader;

dgrv.Columns["D\_IA\_MIN"].Width = clsTaillesGeedoc.tailleDate;

dgrv.Columns["D\_IA\_MAX"].Width = clsTaillesGeedoc.tailleDate;

dgrv.Columns["D\_LN\_MIN"].Width = clsTaillesGeedoc.tailleDate;

dgrv.Columns["D\_LN\_MAX"].Width = clsTaillesGeedoc.tailleDate;

dgrv.Columns["D\_DG\_MIN"].Width = clsTaillesGeedoc.tailleDate;

dgrv.Columns["D\_DG\_MAX"].Width = clsTaillesGeedoc.tailleDate;

dgrv.Columns["D\_MB\_MIN"].Width = clsTaillesGeedoc.tailleDate;

dgrv.Columns["D\_MB\_MAX"].Width = clsTaillesGeedoc.tailleDate;

dgrv.Columns["BD\_SAISIE"].Width = clsTaillesGeedoc.tailleIdUser;

dgrv.Columns["D\_CREATION"].Width = clsTaillesGeedoc.tailleDate;

dgrv.Columns["ID\_CREATEUR"].Width = clsTaillesGeedoc.tailleIdUser;

#endregion

}

else

{

#region Taille liée à la valeur de la Colonne

for (int iCol = 0; iCol < dgrv.ColumnCount; iCol++)

{

dgrv.Columns[iCol].AutoSizeMode = DataGridViewAutoSizeColumnMode.AllCellsExceptHeader;

}

#endregion

}

// En fonction des choix de l’utilisateur afficher ou pas les colonnes

#region Dates DG

if (chkLister\_Afficher\_DatesDG.Checked)

{

dgrv.Columns["D\_DG\_MIN"].Visible = true;

dgrv.Columns["D\_DG\_MAX"].Visible = true;

}

else

{

dgrv.Columns["D\_DG\_MIN"].Visible = false;

dgrv.Columns["D\_DG\_MAX"].Visible = false;

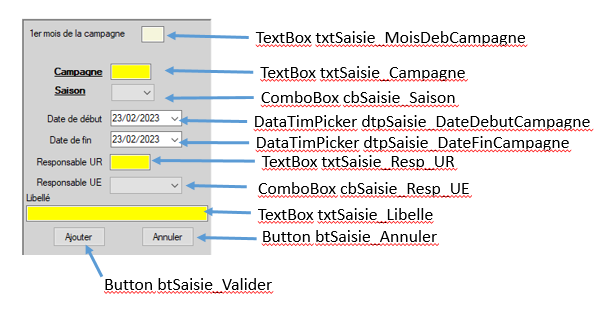
}

#endregion

}

# CRUD : Create, Update, Delete

Le Panel pour Ajouter/Modifier une campagne/saison de reproduction se présente comme ceci :



txtSaisie\_MoisdebCampagne : est initialisé au début en fonction u paramètre du troupeau (non modifiable)

txtSaisie\_Campagne : à saisir

cbSaisie\_Saison : à choisir (fait en conception (de 1 à 6))

dtpSaisie\_DateDebutCampagne : calculé en fonction de Campagne/Saison et du paramètre txtSaisie\_MoisDebCampagne

dtpSaisie\_DateFinCampagne : à saisir ()

txtSaisir\_Resp\_UR : à saisir

cbSaisir\_resp\_UE : à choisir dans une liste

txtSaisie\_Libelle : à saisir

btSaisie\_Valider : va prendre la valeur « ***Ajouter*** » si mode Ajout ou la valeur « ***Modifier*** »

btSaisie\_Annuler : permet d’annuler l’opération et de revenir au mode « ***Ajout*** »

private void btSaisie\_Valider\_Click(object sender, EventArgs e)

{

string operation = btSaisie\_Valider.Text; //Ajouter ou Modifier

string message = "";

int iLigne = 0;

bool continuer = true;

string campagne = "", saison = "", strLibelle="", strDateDebut = "", strDateFin = "", strResp\_UR="", strResp\_UE = "";

if (Campagne\_Verifier\_Champs())

{

if (Campagne\_Verifier\_Dates())

{

campagne = txtSaisie\_Campagne.Text.Trim();

saison = cbSaisie\_Saison.Text.Trim();

strLibelle = txtSaisie\_Libelle.Text.Trim();

strDateDebut = dtpSaisie\_DateDebutCampagne.Value.ToString();

strDateFin = dtpSaisie\_DateFinCampagne.Value.ToString();

strResp\_UR = txtSaisie\_Resp\_UR.Text.Trim();

strResp\_UE = cbSaisie\_Resp\_UE.Text.Trim();

DefCampagne camp\_add = new DefCampagne();

camp\_add.Campagne = campagne;

camp\_add.Saison = saison;

camp\_add.LibelleCampagne = strLibelle;

camp\_add.ResponsableUe = strResp\_UE;

camp\_add.ResponsableUr = strResp\_UR;

camp\_add.DateDebutCampagne = dtpSaisie\_DateDebutCampagne.Value;

camp\_add.DateFinCampagne = dtpSaisie\_DateFinCampagne.Value;

switch (operation) {

case "Ajouter":

continuer = CampagneDao.TableCampagne\_Ajouter\_UnEnregistrementV2(ora\_gen, camp\_add);

if (!continuer) {

message="L'ajout a échoué ! La campagne/saison doit être UNIQUE !";

MessageBox.Show(message, "Campagne-Ajouter", MessageBoxButtons.OK, MessageBoxIcon.Error); }

break;

case "Modifier":

continuer = CampagneDao.TableCampagne\_Modifier\_UnEnregistrementV2(ora\_gen, camp\_add);

if (!continuer) {

message = "La modification a échoué !";

MessageBox.Show(message, "Campagne-Modifier", MessageBoxButtons.OK, MessageBoxIcon.Error) ; }

break;

default:

break;

} // Fin du switch

if (continuer == true)

{

// Comment se repositionner sur celui qui a été modifié

if (operation == "Modifier")

iLigne = dgrvCampagne.CurrentRow.Index;

Campagne\_RechargerDonneesV2();

if (operation == "Modifier") clsDataGridViewHelper.SePositionner\_Sur\_Enregistrement(dgrvCampagne, iLigne);

}

if (continuer)

Campagne\_ReInitialiser\_Champs();

}

}

}

# Côté Base de Données

Normalement toutes les classes sont sur le répertoire DataGeedoc

DefCampagne.cls

Toutes les instructions pour Insérer, Modifier, Supprimer sont sur le répertoire DAO

CampagneDao.cls

## Exemple CampagneDao.cls

## Compter les enregistrements

Select count(\*) FROM … LEFT JOIN .. WHERE ..

Long nb\_enreg = ora\_gen.SQLCount(sql) ;

/// <summary>

/// Permet de vérifier qu'il n'y a pas de reproducteur dans cette campagne/saison

/// </summary>

/// <param name="campagne">Campagne de reproduction</param>

/// <param name="saison">Saison de reproduction</param>

/// <returns>True si ok, false sinon</returns>

private bool Campagne\_Verifier\_Animaux(string campagne, string saison)

{

bool ok = false;

string sql="";

long nb\_enreg = 0;

sql = "SELECT count(\*) FROM " + dbNameDataBase + ".repro R";

sql+=" LEFT JOIN "+dbNameDataBase+".present PR ON R.present\_id=PR.present\_id";

sql+=" LEFT JOIN "+dbNameDataBase+".campagne CA ON R.camprepro\_id=CA.camprepro\_id";

sql += " WHERE PR.troupeau\_id =" + clsVariablesGlobalesGeedoc.myTroupeauId;

sql += " AND CA.campagne ='" + campagne + "'";

sql += " AND CA.saison ='" + saison + "'";

nb\_enreg = ora\_gen.SQLCount(sql);

if (nb\_enreg == 0)

ok = true;

else

ok = false;

return ok;

}

## Insert INTO, UPDATE, DELETE

Les Insert, Update et Delete sont dans <nomTable>DAO

**INSERT** : Syntaxe INSERT INTO .. VALUES (….)

Se fait avec un ExecuteNonQuery()

/// <summary>

/// Permet d'ajouter une campagne/saison dans la table CAMPAGNE

/// </summary>

/// <param name="ora\_gen">Pour accéder à la Base de Données</param>

/// <param name="camp\_add">Campagne/saison à ajouter</param>

/// <returns>Vrai si Ok Sinon renvoie Faux</returns>

static public bool TableCampagne\_Ajouter\_UnEnregistrementV2(clsDataBase ora\_gen, DefCampagne camp\_add)

{

string sql = "";

sql = "INSERT INTO " + clsVariablesGlobalesGeedoc.dbNameDataBase + ".campagne";

sql += "(troupeau\_id,campagne,saison,lib\_campagne,d\_debut,d\_fin,id\_resp\_ur,id\_resp\_ue";

sql +=",d\_creation,createur\_id,bd\_saisie)";

sql += " VALUES (" + clsVariablesGlobalesGeedoc.myTroupeauId;

sql += ",'" + camp\_add.Campagne + "','" + camp\_add.Saison + "','" + camp\_add.LibelleCampagne + "'";

sql += "," + clsSAGAMCBHelper.toDateFormat(camp\_add.DateDebutCampagne, clsSAGAMCBTypesDataGen.TYP\_DATA\_DATE, clsVariablesGlobalesGeedoc.iDB\_USED);

sql += "," + clsSAGAMCBHelper.toDateFormat(camp\_add.DateFinCampagne, clsSAGAMCBTypesDataGen.TYP\_DATA\_DATE, clsVariablesGlobalesGeedoc.iDB\_USED);

// Responsable UR

if (camp\_add.ResponsableUr == "")

sql += ",null";

else

sql += ",'" + camp\_add.ResponsableUr.ToUpper() + "'";

// Responsable UE

if (camp\_add.ResponsableUe == "")

sql += ",null";

else

sql += ",'" + camp\_add.ResponsableUe + "'";

sql += "," + clsSAGAMCBHelper.toDateFormat(DateTime.Today, clsSAGAMCBTypesDataGen.TYP\_DATA\_DATE, clsVariablesGlobalesGeedoc.iDB\_USED);

sql += "," + clsVariablesGlobalesGeedoc.myUserId;

// Pour Dire ajouté dans Geedoc

sql += ",'G')";

if (ora\_gen.ExecuteNonQuery(sql))

return true;

else

return false;

}

**UPDATE** : Syntaxe UPDATE table SET … WHERE …

/// <summary>

/// Modifier la définition d'une Campagne/Saison

/// </summary>

/// <param name="ora\_gen">Pour accéder à la Base de Données</param>

/// <param name="camp\_maj">Campagne/Saison à modifier</param>

/// <returns>Vrai si la modification s'est bien passée sinon renvoie Faux</returns>

static public bool TableCampagne\_Modifier\_UnEnregistrementV2(clsDataBase ora\_gen, DefCampagne camp\_maj)

{

string sql = "";

sql = "UPDATE " + clsVariablesGlobalesGeedoc.dbNameDataBase + ".campagne";

sql += " SET lib\_campagne='" + camp\_maj.LibelleCampagne + "'";

sql += ",d\_debut=" + clsSAGAMCBHelper.toDateFormat(camp\_maj.DateDebutCampagne, clsSAGAMCBTypesDataGen.TYP\_DATA\_DATE, clsVariablesGlobalesGeedoc.iDB\_USED);

sql += ",d\_fin=" + clsSAGAMCBHelper.toDateFormat(camp\_maj.DateFinCampagne, clsSAGAMCBTypesDataGen.TYP\_DATA\_DATE, clsVariablesGlobalesGeedoc.iDB\_USED);

// Responsable UR

if (camp\_maj.ResponsableUr == "")

sql += ", id\_resp\_ur=null";

else

sql += ", id\_resp\_ur='" + camp\_maj.ResponsableUr.ToUpper() + "'";

// Responsable UE

if (camp\_maj.ResponsableUe == "")

sql += ", id\_resp\_ue=null";

else

sql += ", id\_resp\_ue='" + camp\_maj.ResponsableUe + "'";

// Where : je n’utilise pas camprepro\_id (à réfléchir)

sql += " WHERE troupeau\_id =" + clsVariablesGlobalesGeedoc.myTroupeauId;

sql += " AND campagne ='" + camp\_maj.Campagne + "'";

sql += " AND saison ='" + camp\_maj.Saison + "'";

if (ora\_gen.ExecuteNonQuery(sql))

return true;

else

return false;

}

**DELETE** : syntaxe DELETE FROM .. WHERE ..

/// <summary>

/// Permet de supprimer une campagne/saison

/// </summary>

/// <param name="ora\_gen">Pour accéder à la Base de Données</param>

/// <param name="camp\_del">Campagne/saison à supprimer</param>

/// <returns>Vrai si la suppression s'est bien passée sinon Faux</returns>

static public bool TableCampagne\_Supprimer\_UnEnregistrementV2(clsDataBase ora\_gen, DefCampagne camp\_del)

{

string sql = "";

sql = "DELETE FROM " + clsVariablesGlobalesGeedoc.dbNameDataBase + ".campagne";

// Where : je n’utilise pas camprepro\_id (à réfléchir)

sql += " WHERE troupeau\_id =" + clsVariablesGlobalesGeedoc.myTroupeauId;

sql += " AND campagne ='" + camp\_del.Campagne + "'";

sql += " AND saison ='" + camp\_del.Saison + "'";

if (ora\_gen.ExecuteNonQuery(sql))

return true;

else

return false;

}

# Créer des listes d’objets

On utilise sql et la fonction SQLResult\_in\_Array qui met le résultat du SELECT dans un tableau d’objets.

/// <summary>

/// Permet de créer la liste des campagnes pour Un élevage, une espèce, UNE production

/// </summary>

/// <param name="ora\_gen">Pour accéder à la Base de Données</param>

/// <param name="maListeGen">Ma liste à créer de classe Campagne</param>

static public void TableCampagne\_creerListeCampagneV2(clsDataBase ora\_gen, List<DefCampagne> maListeGen)

{

int nb\_ligne = 0;

int nb\_colonnes = 6;

// On fait la liste

maListeGen.Clear();

string sql = "";

sql = "SELECT campagne,saison,lib\_campagne,d\_debut,d\_fin,camprepro\_id";

sql += " FROM " + clsVariablesGlobalesGeedoc.dbNameDataBase + ".campagne";

sql += " WHERE troupeau\_id=" + clsVariablesGlobalesGeedoc.myTroupeauId;

// Trié par

sql += " ORDER BY campagne,saison";

Object[,] tablCamp;

tablCamp = ora\_gen.SQLResult\_in\_Array(sql);

if (tablCamp.Length > 0)

{

nb\_ligne = tablCamp.Length / nb\_colonnes;

for (int iLigne = 0; iLigne < nb\_ligne; iLigne++)

{

DefCampagne camp = new DefCampagne();

camp.Campagne = tablCamp[iLigne, 0].ToString();

camp.Saison = tablCamp[iLigne, 1].ToString();

camp.LibelleCampagne = tablCamp[iLigne, 2].ToString();

if (tablCamp[iLigne, 3].ToString().Trim() != "")

camp.DateDebutCampagne = (DateTime)tablCamp[iLigne, 3];

if (tablCamp[iLigne, 4].ToString().Trim() != "")

camp.DateFinCampagne = (DateTime)tablCamp[iLigne, 4];

camp.Camprepro\_Id = int.Parse(tablCamp[iLigne, 5].ToString());

// Ajouter à la liste des Définitions de Campagne de Reproduction

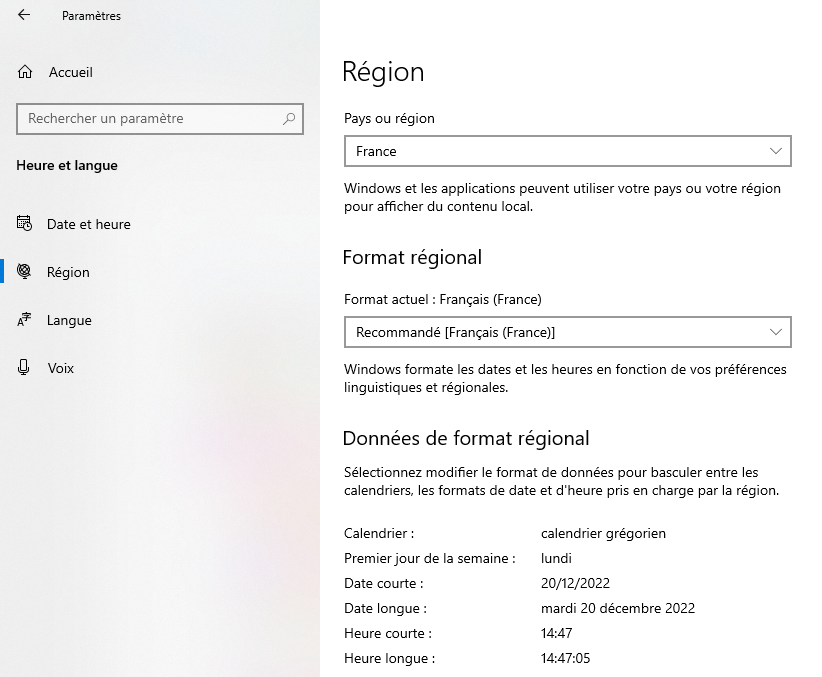
maListeGen.Add(camp);

}

}

}

## Traitement des Dates



Il y a plusieurs Type de Date

Les types de données sont définies dans clsSAGAMCBType qui est dans le namespace SAGAMCBHelpers

public static class clsSAGAMCBTypesDataGen

{

public const int TYP\_DATA\_CHAR = 0;

public const int TYP\_DATA\_INTEGER = 1;

public const int TYP\_DATA\_NUM = 2;

public const int TYP\_DATA\_DATE = 3;

public const int TYP\_DATA\_DATE\_HEURE = 4;

public const int TYP\_DATA\_DATE\_HEURE\_MINUTE = 5;

public const int TYP\_DATA\_DATE\_HEURE\_MINUTE\_SECONDE = 6;

Dans le SQL : Exemple UPDATE

sql += ",d\_debut=" + clsSAGAMCBHelper.toDateFormat(camp\_maj.DateDebutCampagne, clsSAGAMCBTypesDataGen.TYP\_DATA\_DATE, clsVariablesGlobalesGeedoc.iDB\_USED);

C’est la fonction clsSAGAMCBHelper**.toDateFormat** qui va utiliser la bonne fonction liée au SGBD pour traiter les dates (**to\_date** pour Oracle, **to\_timestamp** pour PostgreSql)

Si clsVariablesGlobalesGeedoc.iDB\_USED== clsSAGAMCBTypesDataGen.TYP\_DB\_ORACLE

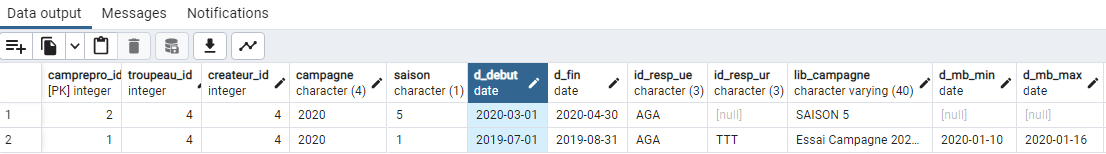
s = "to\_date('" + dt\_gen.Day + "/" + dt\_gen.Month + "/" + dt\_gen.Year + ":00:00:00','DD/MM/YYYY:HH24:MI:SS')";

Si clsVariablesGlobalesGeedoc.iDB\_USED== clsSAGAMCBTypesDataGen.TYP\_DB\_POSTGRESQL

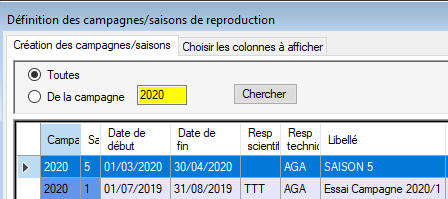
s = "to\_timestamp('" + dt\_gen.Day + "/" + dt\_gen.Month + "/" + dt\_gen.Year + ":00:00:00','DD/MM/YYYY:HH24:MI:SS')";

Affichage des Dates :

Mêmes si dans PgAdmin les dates s’affichent comme au format YYYY-MM-DD



Dans Geedoc, les dates vont s’afficher au format DD/MM/YYYY (Format Date Courte du Paramètre Windows)



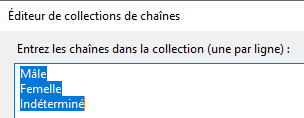
# Les ComboBox

## ComboBox Sexe

Il y a des ComboBox créé à la conception.

Exemple

cbMAJ\_Sexe



Comme les codes pour le Sexe

1=Male

2=Femelle

3=Indéterminé

Il suffit ensuite de travailler sur

int sexe = cbMAJ\_Sexe.SelectedIndex + 1;

string strSexe = sexe.ToString();

SelectIndex commence à 0, il faut donc ajouter 1.

Rappel des codes : Sexe=1 Mâle, Sexe=2 Femelle, Sexe=3 Indéterminé

## ComboBox avec des données simples

Exemple construire un ComboBox avec les Saisons intra Campagne de reproduction

/// <summary>

/// Permet de construire les comboBox sur les saisons

/// </summary>

/// <param name="ora\_gen">Pour accéder à la Base de Données</param>

/// <param name="cbSaison\_Gen">ComboBox contenant les saisons</param>

/// <param name="campagne">Campagne concernée</param>

static public void TableCampagne\_ConstruireComboBox\_SaisonPG2(clsDataBase ora\_gen, ComboBox cbSaison\_Gen, string campagne)

{

string sql = "";

sql = "SELECT DISTINCT saison";

sql += " FROM " + clsVariablesGlobalesGeedoc.dbNameDataBase + ".campagne";

sql += " WHERE troupeau\_id=" + clsVariablesGlobalesGeedoc.myTroupeauId;

sql += " AND campagne='" + campagne + "'";

sql += " ORDER BY saison";

Object[,] tablSaison;

tablSaison = ora\_gen.SQLResult\_in\_Array(sql);

cbSaison\_Gen.Items.Clear();

if (tablSaison.Length > 0)

{

for (int i = 0; i < tablSaison.Length; i++)

{

cbSaison\_Gen.Items.Add(tablSaison[i, 0].ToString());

}

cbSaison\_Gen.SelectedIndex = 0; // Se positionner sur la 1ère saison

}

}

## ComboBox avec un ListItem

La définition du ListItem (objet créé par Hervé Lagant)

public struct ListItem{

public string LabelString;

public string Constant;

public ListItem(string labelString, string constant)

{

this.LabelString = labelString;

this.Constant = constant;

}

public override string ToString {

return LabelString;

}

}

### Exemple pour les Lots de Lutte intra Campagne/Saison

/// <summary>

/// Permet de construire les comboBox sur les Lots de Lutte

/// </summary>

/// <param name="ora\_gen">Pour accéder à la Base de Données</param>

/// <param name="cbLotLutte\_Gen">ComboBox à constuire qui va contenir les Lots de Lutte</param>

/// <param name="campagne">Campagne de reproduction concernée</param>

/// <param name="saison">Saison de reproduction concernée</param>

/// <param name="avec\_lot\_vide">True = ajouter "" au début, false sinon</param>

static public void TableLotLutte\_ConstruireComboBox\_LotLuttePG2(clsDataBase ora\_gen, ComboBox cbLotLutte\_Gen, string campagne, string saison, bool avec\_lot\_vide)

{

string sql = "SELECT LL.lot\_lutte,LL.lotlutte\_id";

sql += " FROM " + clsVariablesGlobalesGeedoc.dbNameDataBase + ".lot\_lutte LL";

sql += " LEFT JOIN " + clsVariablesGlobalesGeedoc.dbNameDataBase + ".campagne CA on LL.camprepro\_id=CA.camprepro\_id";

sql += " WHERE troupeau\_id=" + clsVariablesGlobalesGeedoc.myTroupeauId;

sql += " AND CA.campagne='" + campagne + "'";

sql += " AND CA.saison='" + saison + "'";

sql += " ORDER BY LL.lot\_lutte";

// On créé un tableau d’objets

Object[,] tablLot;

// On lance la requête qui va mettre le résultat dans le tableau d’objets

tablLot = ora\_gen.SQLResult\_in\_Array(sql);

// Initialiser le nombre de colonnes lues

int nb\_lignes = 0, nb\_colonnes = 2;

// Avant de remplir le ComboBox, le nettoyer, le vider

cbLotLutte\_Gen.Items.Clear();

// Attention, ici Label + Constant

if (avec\_lot\_vide)

cbLotLutte\_Gen.Items.Add(new ListItem("",""));

if (tablLot.Length > 0)

{

// Le nb de lignes = longueur du tableau / nb de colonnes lues

nb\_lignes = tablLot.Length / nb\_colonnes;

for (int i = 0; i < nb\_lignes; i++)

{

// Label + Constant

cbLotLutte\_Gen.Items.Add(new ListItem(tablLot[i, 0].ToString().Trim(), tablLot[i, 1].ToString().Trim()));

}

cbLotLutte\_Gen.SelectedIndex = 0; // Se positionner sur la 1ère lot de lutte

}

}

### Initialiser le ComboBox

Saisir la Campagne, choisir la saison

// Construire le Combobox sur les lots de lutte

clsGestionGeedoc.TableLotLutte\_ConstruireComboBox\_LotLuttePG2(ora\_gen, cbMAJ\_LotLutte, CAMPAGNE, SAISON, false);

### Utilisation : récupérer la valeur

Attention :

* Le lot de lutte est dans la partie LabelString de l’objet ListItem
* L’identifiant du lot de lutte (lotlutte\_id) est dans la partie Constant de l’objet ListItem

strLotLutte = ((ListItem)cbMAJ\_LotLutte.SelectedItem).LabelString;

lotLutte\_id = int.Parse(((ListItem)cbMAJ\_LotLutte.SelectedItem).Constant.ToString());

### Exemple avec une Liste d’objet

Je créé d’abord la Liste des produits chimiques

/// <summary>

/// Permet de créer toute la liste des codes des produits chimiques

/// </summary>

/// <param name="ora\_gen">Pour accéder à la base de données</param>

/// <param name="maListeGen">Ma liste à créer de classe CodesProduitChimique</param>

static public void TableProdChi\_creerListeCodesProduitChimique(clsDataBase ora\_gen, List<CodesProduitChimique> maListeGen)

{

string sql = "", strValeurMin = "", strValeurMax = "";

int nb\_ligne = 0;

int nb\_colonnes = 0;

// On fait la liste des codes des produits chimiques

maListeGen.Clear();

sql = "SELECT CD\_PRODUIT,MARQUE,NOM\_PRODUIT,UNITE,VAL\_MIN,VAL\_MAX";

sql += " FROM " + clsVariablesGlobalesGeedoc.dbNameDataBase + ".prod\_chi";

sql += " ORDER BY CD\_PRODUIT";

// Initialiser le nombre de colonnes lues

nb\_colonnes = 6;

// Créer un tableau d’objets

Object[,] tablCodesProduit;

// On lance la requête qui va mettre le résultat dans le tableau d’objets

tablCodesProduit = ora\_gen.SQLResult\_in\_Array(sql);

if (tablCodesProduit.Length > 0)

{

// Le nb de lignes = longueur du tableau / nb de colonnes lues

nb\_ligne = tablCodesProduit.Length / nb\_colonnes;

for (int iLigne = 0; iLigne < nb\_ligne; iLigne++)

{

// Créer un Code Produit Chimique

CodesProduitChimique cr = new CodesProduitChimique();

cr.Code\_Produit = tablCodesProduit[iLigne, 0].ToString().Trim();

cr.Marque = tablCodesProduit[iLigne, 1].ToString().Trim();

cr.Nom\_Produit = tablCodesProduit[iLigne, 2].ToString().Trim();

cr.Unite = tablCodesProduit[iLigne, 3].ToString().Trim();

strValeurMin = tablCodesProduit[iLigne, 4].ToString().Trim();

if (strValeurMin != "")

cr.Val\_Min = float.Parse(strValeurMin);

strValeurMax = tablCodesProduit[iLigne, 5].ToString().Trim();

if (strValeurMax != "")

cr.Val\_Max = float.Parse(strValeurMax);

// Ajouter à la liste des Produits Chimiques

maListeGen.Add(cr);

}

}

}

Puis créer le ComboBox à partir de cette Liste

/// <summary>

/// Permet de construire le comboBox passé en paramètre en donnant la liste des produits chimiques

/// </summary>

/// <param name="cbCodeGen">ComboBox à construire</param>

/// <param name="maListGen">Liste des produits chimiques de la classe CodesProduitChimique</param>

static public void CodesProduitChimique\_ConstruireComboBox\_ByList(ComboBox cbCodeGen, List<CodesProduitChimique> maListGen)

{

cbCodeGen.Items.Clear();

foreach (CodesProduitChimique prod in maListGen)

{

cbCodeGen.Items.Add(new ListItem(prod.Code\_Produit + "=" + prod.Nom\_Produit, prod.Code\_Produit));

}

// Se positionner sur le premier code

cbCodeGen.SelectedIndex = 0;

}

# Opérations sur les Animaux

C’est-à-dire toutes les mesures sur les Animaux : Pesées, Etat Corporels, Abattage, Endoscopie, Oestrus, Psperme,

Et le suivis des animaux Lot\_Physique, Anim\_Expe, Anim\_loc, Anim\_ModCond, ..etc..

Voir le MCD mais en général, il faut Present\_id , une date de mesure et des informations à mettre à jour spécifiques pour chaque type de table.

## Exemple Note d’Etat Corporels (NEC) : Table ETAT\_CORPOREL

### Structure de la table ETAT\_CORPOREL simplifiée

create table geedoc.etat\_corporel (

etatcorporel\_id integer GENERATED ALWAYS AS IDENTITY,

present\_id integer NOT NULL,

operateur\_id integer NULL,

createur\_id integer,

d\_note\_etat date NOT NULL,

note\_lomb integer NULL,

note\_ster integer NULL,

cd\_etat character(2) NULL,

d\_creation date NULL,

constraint pk\_etatcorporel primary key (etatcorporel\_id),

constraint fk\_etatcorporel\_present foreign key (present\_id) references geedoc.present(present\_id),

constraint fk\_etatcorporel\_operateur foreign key (operateur\_id) references geedoc.utilisateur(user\_id),

constraint fk\_etatcorporel\_createur foreign key (createur\_id) references geedoc.utilisateur(user\_id),

UNIQUE (present\_id, d\_note\_etat)

)

#### Classe EtatCorporel

class EtatCorporel {

private int etatcorporel\_id;

Present animalPresent;

private int operateur\_id;

private DateTime date\_nec;

private int note\_lombaire;

private int note\_sternale;

private String code\_nec;

#### Classe Present

class Present {

private int present\_id;

private int troupeau\_id;

private Animal animalGeedoc;

private DateTime dateArriveeElevage;

private DateTime dateSortieElevage;

private string elevageDestination;

private string commentSortie;

private string codeTransfert;

#### Classe Animal

class Animal {

private int animal\_id;

private string elvNe;

private string idAnimal;

private string ipg;

private string rfid;;

private string sexe;

private string typGen;

### Pour Ajouter/Modifier

En mode saisie : Present\_id est lu dans le DataGridView

private void btMAJ\_Valider\_Click(object sender, EventArgs e)

{

**present\_id** = int.Parse(dgrvSaisie[dgrvSaisie\_col\_present\_id, iLigne].Value.ToString().Trim());

Present anim\_present = new Present();

**anim\_present.Present\_Id = present\_id;**

EtatCorporel nec\_add = new EtatCorporel();

**nec\_add.AnimalPresent = anim\_present; // Lien avec l’objet de classe Present**

nec\_add.Date\_Nec = dtpSaisie\_DateNEC.Value;

if (strNoteLombaire == "")

nec\_add.Note\_Lombaire = int.MinValue;

else

nec\_add.Note\_Lombaire = int.Parse(strNoteLombaire);

if (strNoteSternale == "")

nec\_add.Note\_Sternale = int.MinValue;

else

nec\_add.Note\_Sternale = int.Parse(strNoteSternale);

nec\_add.Code\_Nec = strCodeEtat;

nec\_add.Operateur\_Id = operateur\_id;

if (operation == "Ajouter")

{

continuer = EtatCorporelDao.TableEtatCorporel\_Ajouter\_UnEnregistrementPG(ora\_gen, nec\_add);

}

else

{

// Modifier : il faut etatcorporel\_id que je lis dans le DataGridView

etatcorporel\_id = int.Parse(dgrvSaisie[dgrvSaisie\_col\_etatcorporel\_id, iLigne].Value.ToString());

nec\_add.Etatcorporel\_Id = etatcorporel\_id;

continuer = EtatCorporelDao.TableEtatCorporel\_Modifier\_UnEnregistrementPG(ora\_gen, nec\_add);

}

# Opérations sur les Animaux en Reproduction dans une campagne/saison

Ceci concerne les données liées à la partie Reproduction :

Tables : Repro, Lutte\_Nat (repromale\_id), Plan\_Acc (reprofem\_id,repromale\_id), Saillie, Gestation, Scycle, Hormonal, MiseBas

Il faudra donc récupérer ou trouver l’identifiant repro\_id

## Exemple Diagnostic de Gestation : Table Gestation

### Gestation : Structure de la table Gestation

CREATE TABLE geedoc.gestation

(

gestation\_id integer GENERATED ALWAYS AS IDENTITY,

repro\_id integer NOT NULL,

operateur\_id integer,

createur\_id integer,

d\_diagnostic date NOT NULL,

methode character(1) NULL ,

b\_resultat character(1) NULL ,

nb\_foetus integer NULL,

nb\_viables integer NULL,

rang character(1) NULL,

d\_creation date NULL,

constraint pk\_gestation primary key (gestation\_id),

constraint fk\_gestation\_repro foreign key (repro\_id) references geedoc.repro(repro\_id),

constraint fk\_gestation\_opera foreign key (operateur\_id) references geedoc.utilisateur(user\_id),

UNIQUE (repro\_id,d\_diagnostic)

)

#### Classe Gestation

class Gestation {

private int gestation\_id;

private ReproPG animalRepro;

private DateTime dateGestation;

private string methodeGestation;

private string resultatGestation;

private string rangSaillie;

private int nbFoetus;

private int nbViables;

private int operateur\_id;

#### Classe ReproPG

class ReproPG {

private int repro\_id;

private Present animalPresent;

private DefCampagne campagneRepro;

private LotLutte lotLutteRepro;

private GroupeMiseBas groupeMiseBasRepro;

private DateTime dateDebutSaison;

private DateTime dateDernierDg;

### Pour Ajouter/Modifier

private void btSaisie\_Valider\_Click(object sender, EventArgs e)

{

ReproPG repro\_maj = new ReproPG();

repro\_maj.Repro\_Id = int.Parse(dgrvFemelle[coldgrvFemelle\_reproId, iLigne].Value.ToString());

Gestation dg\_add = new Gestation();

dg\_add.AnimalRepro = repro\_maj; // Lien avec l’objet de la classe ReproPG

dg\_add.DateGestation = dtDateDG;

dg\_add.ResultatGestation = strResultat;

dg\_add.MethodeGestation = strMethode;

// Attention ListItem

dg\_add.Operateur = ((ListItem)cbSaisie\_Operateur.SelectedItem).LabelString;

dg\_add.Operateur\_Id = int.Parse(((ListItem)cbSaisie\_Operateur.SelectedItem).Constant.ToString());

if (operation == "Ajouter")

{

// Faut-il vérifier la date ? OUI

continuer = Saisie\_Verifier\_DateDG(iLigne, message);

if (continuer)

{

continuer = GestationDao.TableGestation\_Ajouter\_UnEnregistrement(ora\_gen, dg\_add);

}

}

else if (operation == "Modifier")

{

// Il faut gestation\_id

dg\_add.Gestation\_Id = int.Parse(txtSaisie\_GestationID.Text);

// Pas besoin de vérifier la date car elle est NON modifiable

continuer = GestationDao.TableGestation\_Modifier\_UnEnregistrement(ora\_gen, dg\_add);

}