Aide au Développement sous Visual Studio

Fait le 06/03/2024 MC Batut

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

|  |  |  |
| --- | --- | --- |
|  | 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 | lstAfficher |
| 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 frmCampagne

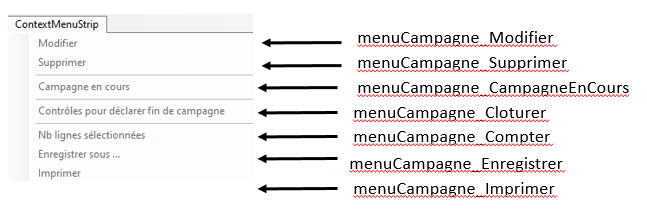


## Lien entre DataGridView et ContexMenuStrip

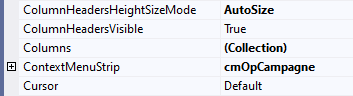
Un DataGridView : dgrvCampagne

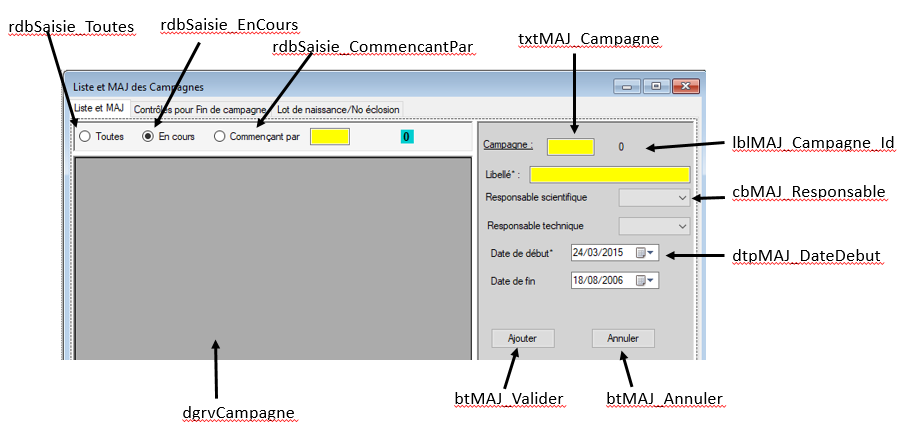
Un ContextMenuStrip : cmOpCampagne





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

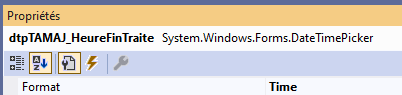


DateTimePicker : mettre format propriété Format Short pour avoir la date format DD/MM/YYYY



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





DataGridView : mettre la plupart du temps propriété SelectionMode = FulRowSelect

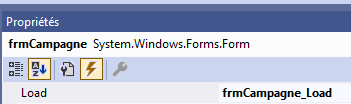


textBox : mettre un MaxLength et mettre en Jaune si à Saisir ou Jaune clair Non à Saisir

# 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 clsDimGenSivol

txtCampagne.MaxLength = clsDimGenSivol.dimCampagne;

txtCampagneMAJ\_Id.MaxLength = clsDimGenSivol.dimCampagne;

txtCampagneMAJ\_Libelle.MaxLength = clsDimGenSivol.dimLibelle;

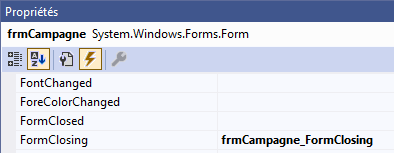
// Onglet Lot Naissance

txtLotNais\_Annee.MaxLength = clsDimGenSivol.dimAnneeNais;

Initialisations des comboBox,

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

**Dans le Closing :**



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

{

ds\_gen.Dispose();

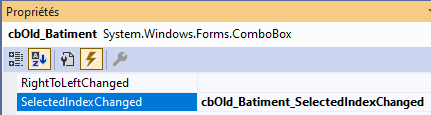
ora\_gen.bdClose();

ora\_gen.bdDispose();

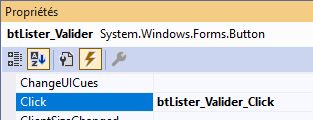
GC.Collect();

}

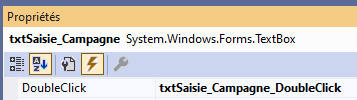
**Pour un objet ComboBox :** selectIndexChanged



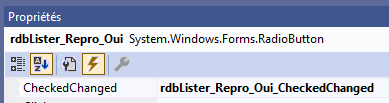
**Pour un objet boutton :** click



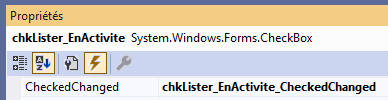
**Pour un objet textBox :** DoubleClick ou un Click



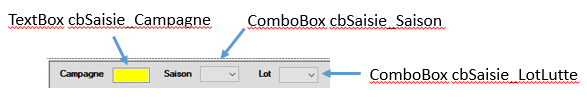
**Pour un objet radioButton :** ChechedChanged



**Pour un objet checkBox :** CheckedChanged



**Exemple : pour des saisies liées à la Ponte**, il faudra Campagne, le groupe de ponte et le numéro de collecte d’oeufs



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

/// <summary>

/// Double clic sur la campagne : permet de lister les groupes de ponte,

/// puis les collectes d'oeufs intra groupe de ponte

/// </summary>

/// <param name="sender"></param>

/// <param name="e"></param>

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

{

string message = "";

txtSaisie\_Campagne.Text = clsSAGAMCBHelper.AjouterZeros(txtSaisie\_Campagne.Text, dimCampagne);

SAISIR\_CAMPAGNE = txtSaisie\_Campagne.Text;

// Créer la liste des groupes de pontes en partant des collecte d'oeufs : il faut avoir saisi l'incubation

clsGestionSivol.TableCollecteOeuf\_Creer\_ListeGroupePonte(ora\_gen, SAISIR\_CAMPAGNE, maListeGroupePonte, clsConstSivol.COLLECTE\_INCUBATION);

if (maListeGroupePonte.Count ==0)

{

message = "Définir d'abord les collecte d’oeufs pour cette campagne !";

MessageBox.Show(message, "Mirages", MessageBoxButtons.OK, MessageBoxIcon.Error);

cbSaisie\_GroupePonte.Enabled = false;

cbSaisie\_NoCollecte.Enabled = false;

}

else

{

clsGestionSivol.TableGroupePonte\_ConstruireComboBox\_ByList(cbSaisie\_GroupePonte, maListeGroupePonte);

cbSaisie\_GroupePonte.Enabled = true;

cbSaisie\_GroupePonte\_SelectedIndexChanged(sender, e);

}

}

Et quand l’utilisateur va choisir le groupe de ponte, on va construire le ComboBox lié aux collectes d’œufs intra campagne et groupe de ponte

/// <summary>

/// Permet de construire la liste des collectes quand on change de groupe de ponte

/// </summary>

/// <param name="sender"></param>

/// <param name="e"></param>

private void cbSaisie\_GroupePonte\_SelectedIndexChanged(object sender,System.EventArgs e)

{

// Groupe de ponte

SAISIR\_GRP\_PONTE = ((ListItem)cbSaisie\_GroupePonte.SelectedItem).LabelString;

clsGestionSivol.TableCollecteOeuf\_Creer\_ListeCollecteOeuf(ora\_gen, SAISIR\_CAMPAGNE, SAISIR\_GRP\_PONTE, maListeCollecteOeuf, clsConstSivol.COLLECTE\_MIRAGES);

if (maListeCollecteOeuf.Count == 0)

{

cbSaisie\_NoCollecte.Enabled = false;

}

else

{

cbSaisie\_NoCollecte.Enabled = true; clsGestionSivol.TableCollecteOeuf\_ConstruireComboBox\_ByList(cbSaisie\_NoCollecte, maListeCollecteOeuf);

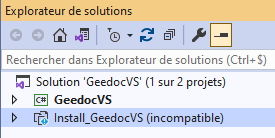
}

}

# Organisation pour Sivol

## L’application Sivol : Ici SivolPG

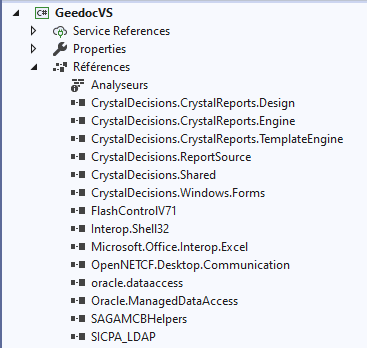
Il y a 1 solution et 2 projets mais ici mon projet pour déployer a un Pb)



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

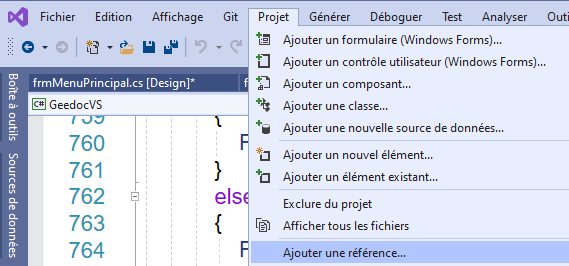
1 projet pour le déploiement (Pour faire le SetUp.exe : ici il y a un Pb)

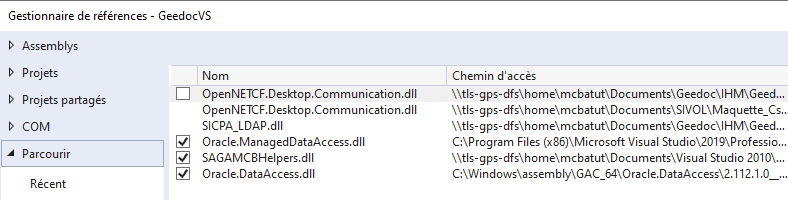
## Ajout de Dll ou de Référence



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

Pour Ajouter une DLL.



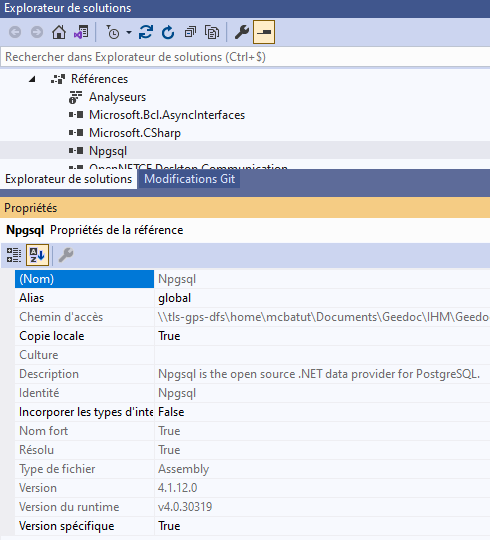


Cliquer sur « **Parcourir** » puis aller chercher la Dll.

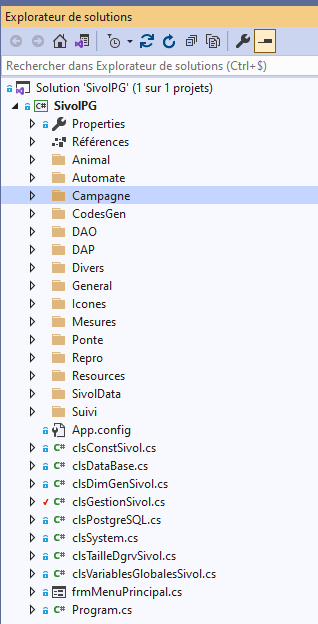
### Pour accéder à la Base de Données

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)

clsPostgreSQL.cs : fonctions spécifiques pour atteindre une base PostgreSql

clsConstSivol.cs : contient toutes les constantes

// Les Espèces

public const int ESPECE\_CANARD = 6;

public const int ESPECE\_OIE = 7;

public const int ESPECE\_POULE = 8;

public const int ESPECE\_CAILLE = 9;

public const int ESPECE\_DINDE = 10;

public const int ESPECE\_PINTADE = 11;

public const int ESPECE\_FAISAN = 12;

clsDimGenSivol.cs : dimension générale utilisé pour les TextBox

static public int dimAnneeNais = 4 ;

static public int dimBatiment = 4;

static public int dimBooleen = 1; // Booléen O/N

static public int dimCage = 4;

static public int dimCampagne = 5;

static public int dimCellule = 3;

clsTaillesDgrvSivol.cls : taille des colonnes dans les DataGridView

static public int tailleAnneeNais = 50;

static public int tailleBatiment = 50;

static public int tailleBoite = 50;

static public int tailleBooleen = 30;

static public int tailleCage = 50;

static public int tailleCampagne = 50;

static public int tailleCellule = 40;

clsVariablesGlobalesSivol.cs : variables globales

static public string myIdUserLdap = "";

static public string myIdUserBD = ""; // New Pour PostgreSQL

static public string myIdUser = "";

static public string myPassword = "";

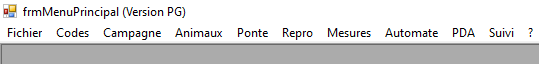
static public string myEspece = "";

clsGestionSivol.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

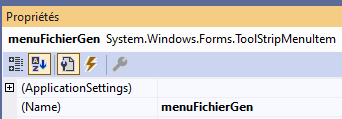
## Les Menus

**Forme Principale : frmMenuPrincipal**





menuGen : Menu

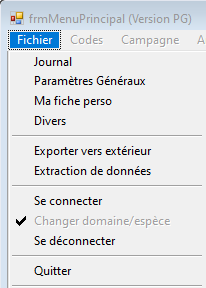


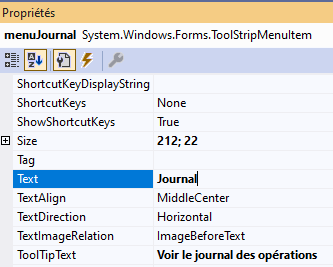
Timer :

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

Propriété ToolTipText

menuFichier





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;

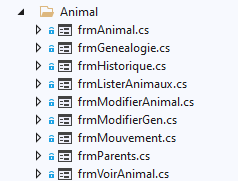
Fille.Show();

}

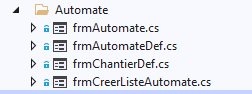
}

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

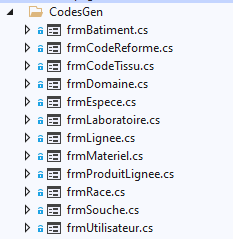
**Répertoire Animal**



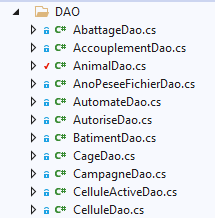
**Répertoire Automate**



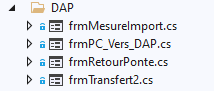
**Répertoire CodesGen**



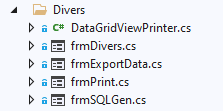
**Répertoire DAO : Data Access Objet (Insert/Update/Delete)**

****

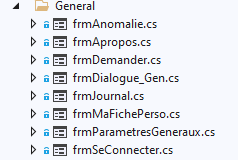
**Répertoire DAP**



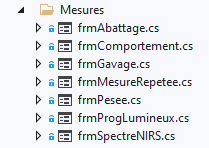
**Répertoire Divers**

****

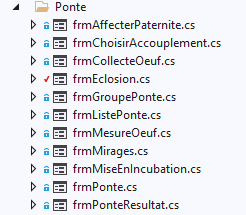
**Répertoire Général**

****

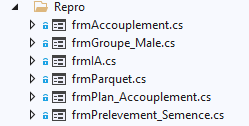
**Répertoire Mesures**



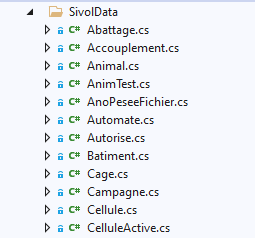
**Répertoire Ponte**



**Répertoire Repro**



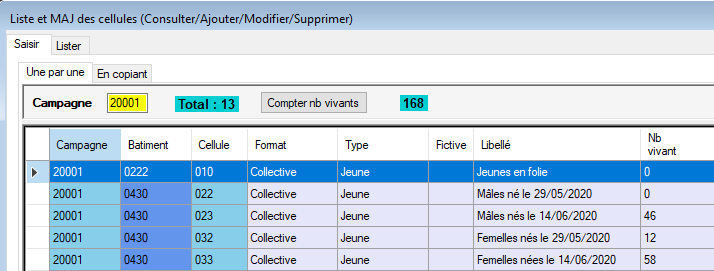
**Répertoire SivolData :** Toutes les classes d’objets liées aux tables de données

****

**Répertoire Suivi**

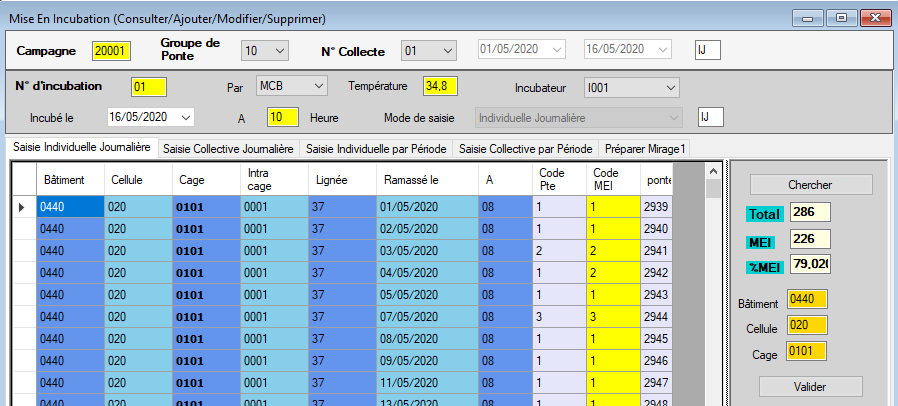
****

# Convention des couleurs dans Geedoc



En Jaune : A saisir

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



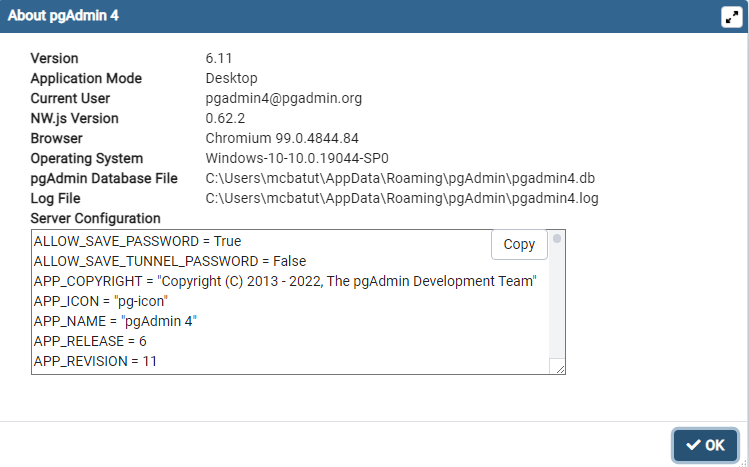
En vert : les totaux = nombre de lignes affichées

En Orange : pour aller se positionner directement sur une cage.

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

txtSaisie\_Campagne.BackColor = clsSAGAMCBCouleur.coulInfoNePasSaisir;

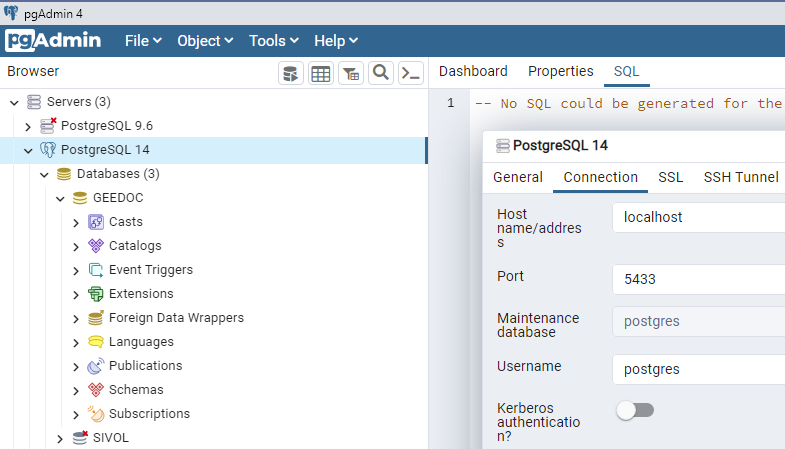
# Lien avec la base de Données



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



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



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

static class clsVariablesGlobalesSivol

{

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

public const string dbDataSource = "host=localhost; port=5433; database=SIVOL";

public const string dbNameDataBase = "sivol";

// public const string dbAdresseIpServeur = "193.54.97.101";

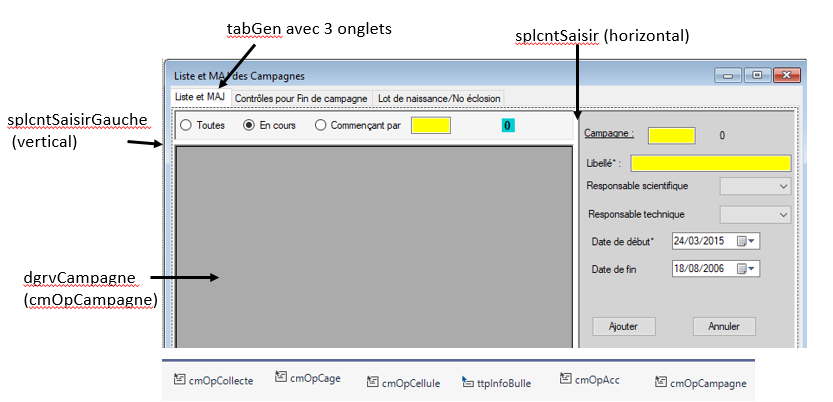
public const uint dbPortBD = 5433; //Pour Sivol PC

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

Les campagnes s’affichent dans le load de la forme ou quand on clique sur les radiobuton « Toutes , «  En cours » ou «  Commençant par »

La procédure Campagne\_RechargerDonnéees va lire les données et les afficher dans le DataFridView dgrvCampagne

private void frmCampagne\_Load(object sender, System.EventArgs e)

{

// Initialiser les dimensions pour les TextBox

Ini\_Dimension\_Generales();

// Afficher les données sur les campagnes

Campagne\_RechargerDonnees();

….

}

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 Campagne

/// <summary>

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

/// </summary>

private void Campagne\_RechargerDonnees()

{

string sql = "";

sql = "SELECT CAM.campagne,UT1.id\_user as id\_responsable,UT2.id\_user as id\_resp\_tech,CAM.bd\_historique" +

",CAM.d\_debut,CAM.d\_fin,CAM.avec\_ponte,CAM.lib\_campagne,CAM.campagne\_id,CAM.responsable\_id,CAM.resptech\_id";

sql += " FROM sivol.campagne CAM ";

sql += " LEFT JOIN sivol.utilisateur UT1 ON CAM.responsable\_id=UT1.user\_id";

sql += " LEFT JOIN sivol.utilisateur UT2 ON CAM.resptech\_id=UT2.user\_id";

// Where

sql += " WHERE CAM.troupeau\_id= " + clsVariablesGlobalesSivol.myTroupeauId;

if (rdbCampagne\_EnCours.Checked)

sql += " AND CAM.d\_fin is null";

else if (rdbCampagne\_CommencantPar.Checked)

sql += " AND CAM.campagne like '" + txtCampagne.Text.Trim() + "%'";

sql += " ORDER BY CAM.campagne DESC";

string nomTable = "CAMPAGNE";

Cursor.Current = Cursors.WaitCursor;

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

{

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

dgrvCampagne.ReadOnly = true;

Campagne\_Afficher\_TitreColonnes(dgrvCampagne);

// Afficher Total

lblTotal.Text = "Total : " + dgrvCampagne.RowCount.ToString();

lblTotal.BackColor = clsSAGAMCBCouleur.coulTotal;

if (dgrvCampagne.RowCount == 0)

{

cmOpCampagne.Enabled = false;

}

else

{

cmOpCampagne.Enabled = true;

menuCampagne\_CampagneEnCours.Enabled = true; // Est-ce que tout le monde on a le droit de modifier Autorise ?

menuCampagne\_CompterSelection.Enabled = true; // on a toujours le droit

menuCampagne\_Enregistrer.Enabled = true; // on a toujours le droit

menuCampagne\_Imprimer.Enabled = true; // on a toujours le droit

menuCampagne\_Modifier.Enabled = myDroit\_Update; // droit UPDATE

menuCampagne\_Supprimer.Enabled = myDroit\_Delete; // droit DELETE

menuCampagne\_FinCampagne.Enabled = myDroit\_Update; // droit UPDATE

}

btCampagneMAJ\_Valider.Enabled = myDroit\_Insert; // droit INSERT

}

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>

/// Afficher le titre des colonnes dans DatagridView à notre convention

/// </summary>

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

private void Campagne\_Afficher\_TitreColonnes(DataGridView dgrv)

{

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

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

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

dgrv.Columns["BD\_HISTORIQUE"].HeaderText = "BD";

dgrv.Columns["D\_DEBUT"].HeaderText = "Début";

dgrv.Columns["D\_FIN"].HeaderText = "Fin";

dgrv.Columns["AVEC\_PONTE"].HeaderText = "Pte";

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

// 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.coulColCle;

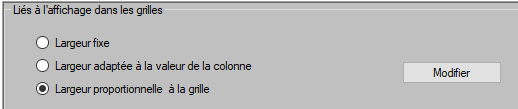
// Dimensionner les colonnes

Campagne\_Redimensionner\_DataGridView(dgrv);

}

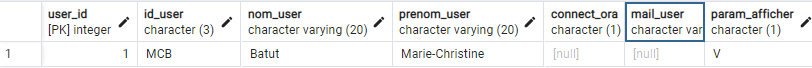
Plusieurs mode d’affichage.

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 dgrvGen

/// </summary>

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

private void Campagne\_Redimensionner\_DataGridView(DataGridView dgrv)

{

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

{

dgrv.RowHeadersWidth = clsSAGAMCBCouleur.tailleLigneEnTete;

if (clsVariablesGlobalesSivol.myParamAfficher == "C")

{

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

{

dgrv.Columns[iCol].AutoSizeMode = DataGridViewAutoSizeColumnMode.AllCellsExceptHeade ; }

}

else if (clsVariablesGlobalesSivol.myParamAfficher == "V")

{

#region Mode Variable

int taille = dgrv.Width - dgrv.RowHeadersWidth - 4;

double taille\_date = 0.18; // 2 \* 18 = 36

double taille\_lot = 0.06; // 4 \* 6 = 24

double taille\_camp = 0.12; // 1 \* 12 = 12

double taille\_libelle = 0.24; // 1 \* 24 = 24

// total = 96

dgrv.Columns["CAMPAGNE"].Width = (int)(taille \* taille\_camp);

dgrv.Columns["ID\_RESPONSABLE"].Width = (int)(taille \* taille\_lot);

dgrv.Columns["ID\_RESP\_TECH"].Width = (int)(taille \* taille\_lot);

dgrv.Columns["BD\_HISTORIQUE"].Width = (int)(taille \* taille\_lot);

dgrv.Columns["D\_DEBUT"].Width = (int)(taille \* taille\_date);

dgrv.Columns["D\_FIN"].Width = (int)(taille \* taille\_date);

dgrv.Columns["AVEC\_PONTE"].Width = (int)(taille \* taille\_lot);

dgrv.Columns["LIB\_CAMPAGNE"].Width = (int)(taille \* taille\_libelle);

#endregion

}

else

{

#region Fixe

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

dgrv.Columns["ID\_RESPONSABLE"].Width = clsTailleDgrvSivol.tailleOperateur;

dgrv.Columns["ID\_RESP\_TECH"].Width = clsTailleDgrvSivol.tailleOperateur;

dgrv.Columns["BD\_HISTORIQUE"].Width = clsTalleDgrvSivol.tailleLot;

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

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

dgrv.Columns["AVEC\_PONTE"].Width = clsTailleDgrvSivol.tailleBooleen;

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

#endregion

}

#region Id PG

dgrv.Columns["CAMPAGNE\_ID"].AutoSizeMode = DataGridViewAutoSizeColumnMode.AllCellsExceptHeader;

dgrv.Columns["RESPONSABLE\_ID"].AutoSizeMode = DataGridViewAutoSizeColumnMode.AllCellsExceptHeader;

dgrv.Columns["RESPTECH\_ID"].AutoSizeMode = DataGridViewAutoSizeColumnMode.AllCellsExceptHeader;

dgrv.Columns["CAMPAGNE\_ID"].Visible = clsVariablesGlobalesSivol.myAfficherIdPg;

dgrv.Columns["RESPONSABLE\_ID"].Visible = clsVariablesGlobalesSivol.myAfficherIdPg;

dgrv.Columns["RESPTECH\_ID"].Visible = clsVariablesGlobalesSivol.myAfficherIdPg;

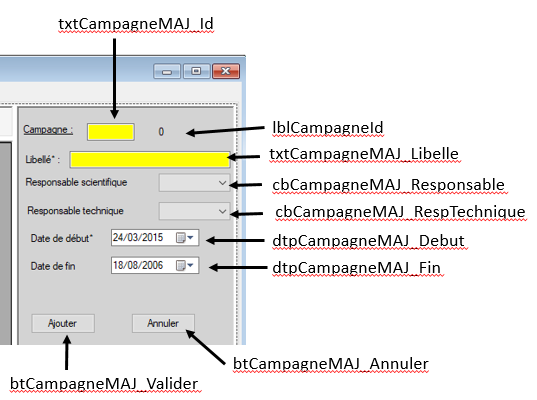
#endregion

}

}

# CRUD : Create, Update, Delete

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



txtCampagneMAJ\_Id : à saisir

lblCampagneID : sera renseigné en mode « Modifier »

dtpCampagneMAJ\_Debut : Date de début à saisir

dtpCampagneMAJ\_Fin : non à saisir pour le moment (Sera à saisir pour Clôturer la campagne))

txtCampagneMAJ\_RespChercheur : à choisir dans la liste

cbCampagneMAj\_RespTechnique: à choisir dans une liste

txtCampagneMAj\_Libelle : à saisir

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

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

/// <summary>

/// Clic sur le bouton btCampagneMAJ\_Valider :

/// C'est à dire soit pour valider un Ajout ou une Modification

/// </summary>

/// <param name="sender"></param>

/// <param name="e"></param>

private void btCampagneMAJ\_Valider\_Click(object sender, System.EventArgs e)

{

string operation = btCampagneMAJ\_Valider.Text;

string message = "";

int iLigneCourrante = 0;

bool continuer = false;

string strCampagne = txtCampagneMAJ\_Id.Text;

string strResponsable = cbCampagneMAJ\_RespChercheur.Text;

string strRespTechnique = cbCampagneMAJ\_RespTechnique.Text;

string strLibelle = txtCampagneMAJ\_Libelle.Text.Trim();

string strDateDebut = dtpCampagneMAJ\_Debut.Value.ToString();

string strDateFin = dtpCampagneMAJ\_Fin.Value.ToString();

if (Campagne\_Verifier\_Champs())

{

Campagne campagne\_add = new Campagne();

campagne\_add.Id\_Campagne = strCampagne;

campagne\_add.Lib\_Campagne = strLibelle;

// Mettre responsable\_id et resptech\_id

campagne\_add.Resp\_Scientifique = strResponsable;

campagne\_add.Resp\_Technique = strRespTechnique;

campagne\_add.D\_Debut\_Campagne = dtpCampagneMAJ\_Debut.Value;

switch (operation)

{

case "Ajouter":

continuer = CampagneDao.TableCampagne\_Ajouter\_UnEnregistrement(ora\_gen, campagne\_add);

if (!continuer)

{

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

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

}

break;

case "Modifier":

// Nouvelle clé PG

campagne\_add.Campagne\_Id = int.Parse(lblCampagneID.Text);

if (Campagne\_Verifier\_Date(DEBUT\_CAMPAGNE))

{

continuer = CampagneDao.TableCampagne\_Modifier\_UnEnregistrement(ora\_gen, campagne\_add);

if (!continuer)

{

message = "La modification de la campagne a échou MessageBox.Show(message, "Campagne", MessageBoxButtons.OK, MessageBoxIcon.Warning);

}

}

break;

case "Valider":

// Clôturer la campagne

campagne\_add.D\_Fin\_Campagne = dtpCampagneMAJ\_Fin.Value;

if (Campagne\_Verifier\_Date(FIN\_CAMPAGNE))

{

continuer = CampagneDao.TableCampagne\_Modifier\_DateFin(ora\_gen, campagne\_add);

if (!continuer)

{

message = "La mise à jour de la date de fin a échoué !";

MessageBox.Show(message, "Campagne", MessageBoxButtons.OK, MessageBoxIcon.Warning);

}

}

break;

default:

break;

} // Fin du switch

if (continuer == true)

{

// Se repositionner sur celui qui a été modifié

if (operation != "Ajouter")

iLigneCourrante = dgrvCampagne.CurrentRow.Index;

Campagne\_RechargerDonnees();

if (operation != "Ajouter")

{

clsDataGridViewHelper.SePositionner\_Sur\_Enregistrement(dgrvCampagne, iLigneCourrante);

}

Campagne\_Initialiser\_Champs\_PourAjout();

}

}

}

# Côté Base de Données

Normalement toutes les classes sont sur le répertoire DataSivol

Campagne.cls

namespace SivolData

{

public class Campagne

{

private int campagne\_id;

private int troupeau\_id;

private int responsable\_id;

private int resptech\_id;

private string id\_campagne;

private string lib\_campagne;

private string resp\_scientifique;

private string resp\_technique;

private DateTime d\_debut\_campagne;

private DateTime d\_fin\_campagne;

..etc..

}

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 un enregistrement ici une campagne dans la table CAMPAGNE

/// </summary>

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

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

/// <returns>Vrai si l'ajout s'est bien passé sinon renvoie faux</returns>

static public bool TableCampagne\_Ajouter\_UnEnregistrement(clsDataBase ora\_gen, Campagne campagne\_add)

{

string sql = "";

sql = "INSERT INTO sivol.campagne (troupeau\_id,campagne,id\_responsable,id\_resp\_tech,d\_debut,lib\_campagne)";

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

sql += ",'" + campagne\_add.Id\_Campagne + "'";

// Il faut responsable\_id

if (campagne\_add.Resp\_Scientifique == "")

sql += ",null";

else

sql += ",'" + campagne\_add.Resp\_Scientifique + "'";

//Il faut mettre resptech\_id

if (campagne\_add.Resp\_Technique == "")

sql += ",null";

else

sql += ",'" + campagne\_add.Resp\_Technique + "'";

sql += "," + clsSAGAMCBHelper.toDateFormat(campagne\_add.D\_Debut\_Campagne, clsSAGAMCBTypesDataGen.TYP\_DATA\_DATE, clsVariablesGlobalesSivol.iDB\_USED);

sql += ",'" + campagne\_add.Lib\_Campagne + "')";

if (ora\_gen.ExecuteNonQuery(sql))

return true;

else

return false;

}

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

/// <summary>

/// Permet de modifier UNE campagne

/// Seuls le libellé et la date de début sont modifiés

/// </summary>

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

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

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

static public bool TableCampagne\_Modifier\_UnEnregistrement(clsDataBase ora\_gen, Campagne campagne\_maj)

{

string sql = "";

sql = "UPDATE sivol.campagne ";

sql += " set lib\_campagne='" + campagne\_maj.Lib\_Campagne + "'";

// Responsable scientifique

if (campagne\_maj.Resp\_Scientifique == "")

sql += " , id\_responsable=null";

else

sql += " , id\_responsable='" + campagne\_maj.Resp\_Scientifique + "'";

// Responsable technique

if (campagne\_maj.Resp\_Technique == "")

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

else

sql += " , id\_resp\_tech='" + campagne\_maj.Resp\_Technique + "'";

sql += ", d\_debut=" + clsSAGAMCBHelper.toDateFormat(campagne\_maj.D\_Debut\_Campagne, clsSAGAMCBTypesDataGen.TYP\_DATA\_DATE, clsVariablesGlobalesSivol.iDB\_USED);

// WHERE mettre campagne\_id

sql += " WHERE campagne='" + campagne\_maj.Id\_Campagne + "'";

sql += " AND troupeau\_id=" + clsVariablesGlobalesSivol.myTroupeauId;

if (ora\_gen.ExecuteNonQuery(sql))

return true;

else

return false;

}

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

/// <summary>

/// Permet de supprimer la campagne passée en paramètre

/// </summary>

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

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

/// <returns>True si la suppression a abouti, false sinon</returns>

static public bool TableCampagne\_Supprimer\_UnEnregistrement(clsDataBase ora\_gen, Campagne campagne\_del)

{

string sql = "";

sql = "DELETE FROM sivol.campagne";

sql += " WHERE campagne='" + campagne\_del.Id\_Campagne + "'";

sql += " AND troupeau\_id=" + clsVariablesGlobalesSivol.myTroupeauId;

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

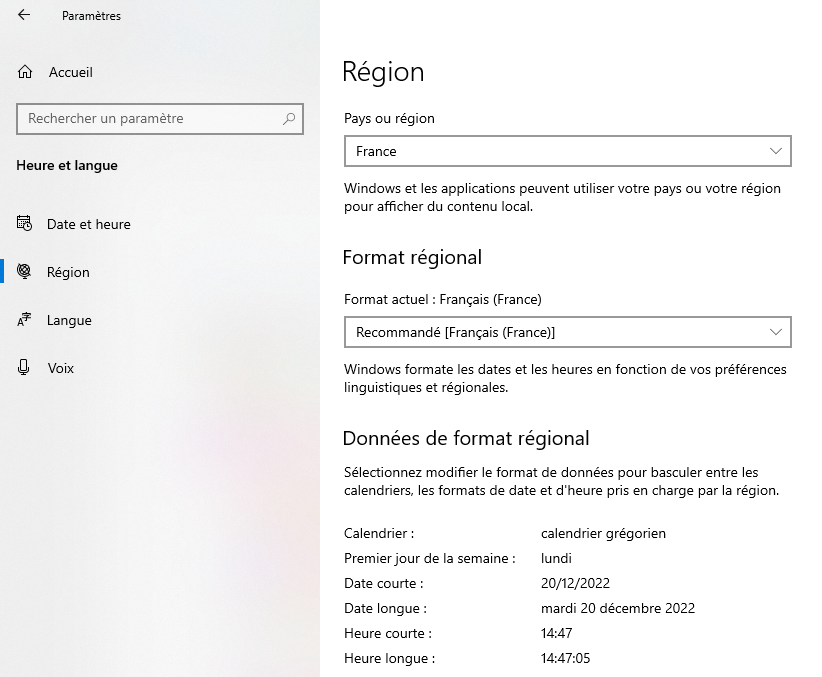
maListeGen.Add(camp);

}

}

}

## 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 clsVariablesGlobalesSivol.iDB\_USED== clsSAGAMCBTypesDataGen.TYP\_DB\_ORACLE

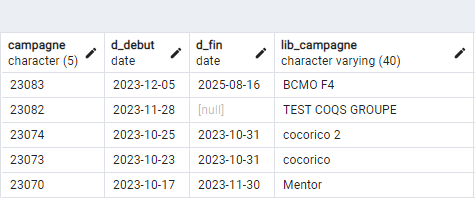
s = "to\_date('" + dt\_gen.Day + "/" + dt\_gen.Month + "/" + dt\_gen.Year + ":00:00:00','DD/MM/YYYY:HH24:MI:SS')";

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

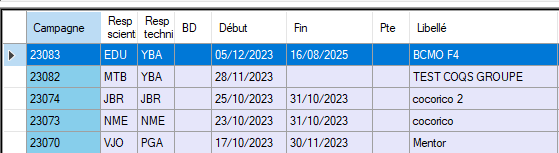
s = "to\_timestamp('" + dt\_gen.Day + "/" + dt\_gen.Month + "/" + dt\_gen.Year + ":00:00:00','DD/MM/YYYY:HH24:MI:SS')";

Affichage des Dates :

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



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



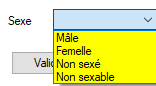
# Les ComboBox

## 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=Non sexé

4=Non sexable

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 Non sexé, 4=Non sexable

## 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 avec une Liste d’objet des groupes de pontes

Je créé d’abord la Liste des groupes de pontes utilisées dans les collecte d’oeufs intra campagne et suivant l’opération demandée

/// <summary>

/// Permet de créer une liste d'objet GroupePonte en fonction de la campagne intra des collecte d'oeufs

/// </summary>

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

/// <param name="strCampagne">Campagne</param>

/// <param name="maListeGoupePonteGen">Liste des groupes de ponte</param>

/// <param name="operation">Opération demandée</param>

static public void TableCollecteOeuf\_Creer\_ListeGroupePonte(clsDataBase ora\_gen, string strCampagne, List<GroupePonte> maListeGoupePonteGen, int operation)

{

int nb\_colonnes = 5, nb\_lignes = 0;

string sql = "";

sql = "SELECT distinct CAM.campagne\_id,GRP.grp\_ponte,CO.groupeponte\_id,GRP.lib\_groupe,GRP.format\_cage";

sql += " FROM sivol.collecte\_oeuf CO";

sql += " INNER JOIN sivol.groupe\_ponte GRP ON CO.groupeponte\_id=GRP.groupeponte\_id";

sql += " INNER JOIN sivol.campagne CAM ON GRP.campagne\_id=CAM.campagne\_id";

sql += " WHERE CAM.troupeau\_id=" + clsVariablesGlobalesSivol.myTroupeauId;

sql += " AND CAM.campagne='" + strCampagne + "'";

// Attention : va dépendre de l'opération demandée

if (operation == clsConstSivol.COLLECTE\_PONTEINDIV)

sql += " AND CO.saisie\_ponte='IJ' ";

else if (operation == clsConstSivol.COLLECTE\_INCUBATION)

sql += " AND CO.d\_fin\_collecte is not null";

else if (operation == clsConstSivol.COLLECTE\_MIRAGES)

sql += " AND CO.no\_incu is not null";

else if (operation == clsConstSivol.COLLECTE\_PATERNITE)

{ sql += " AND CO.no\_incu is not null";

sql += " AND CO.d\_mirage2 is not null";}

else if (operation == clsConstSivol.COLLECTE\_ECLOSION)

sql += " AND CO.aff\_paternite is not null";

sql += " ORDER BY GRP.grp\_ponte";

Object[,] tablGroupe;

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

maListeGoupePonteGen.Clear();

if (tablGroupe.Length > 0)

{

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

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

{

GroupePonte groupe\_maj = new GroupePonte();

groupe\_maj.Campagne = strCampagne;

groupe\_maj.Campagne\_Id = int.Parse(tablGroupe[i, 0].ToString().Trim());

groupe\_maj.Groupe\_Ponte = tablGroupe[i, 1].ToString().Trim();

groupe\_maj.Groupeponte\_Id = int.Parse(tablGroupe[i, 2].ToString().Trim());

groupe\_maj.Lib\_Groupe\_Ponte = tablGroupe[i, 3].ToString().Trim();

groupe\_maj.Format\_Cage = tablGroupe[i, 4].ToString().Trim();

maListeGoupePonteGen.Add(groupe\_maj);

}

}

}

Puis créer le ComboBox à partir de cette Liste

/// <summary>

/// Permet de construire le combobox cbGroupePonte sur les groupes de ponte

/// </summary>

static public void TableGroupePonte\_ConstruireComboBox\_ByList(ComboBox cbGroupePonteGen, List<GroupePonte> maListeGroupePonteGen)

{

cbGroupePonteGen.Items.Clear();

if (maListeGroupePonteGen.Count > 0)

{

foreach (GroupePonte groupe\_maj in maListeGroupePonteGen)

{

// Label + Constant

cbGroupePonteGen.Items.Add(new ListItem(groupe\_maj.Groupe\_Ponte, groupe\_maj.Groupeponte\_Id.ToString()));

}

cbGroupePonteGen.SelectedIndex = 0;

}

}

### Utilisation : récupérer la valeur

Attention :

* Le groupe de ponte est dans la partie LabelString de l’objet ListItem
* L’identifiant du groupe de ponte (groupeponte\_id) est dans la partie Constant de l’objet ListItem

strGroupePonte = ((ListItem)cbSaisie\_GroupePonte.SelectedItem).LabelString;

groupeponte\_id = int.Parse(((ListItem)cbSaisie\_GroupePonte.SelectedItem).Constant.ToString());

# Opérations sur les Animaux

C’est-à-dire toutes les mesures sur les Animaux : Pesées, Abattage, ..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 : Table ABATTAGE

#### Classe abattage

class Abattage

{

private int abattage\_id;

private Present animalPresent;

private DateTime d\_abattage;

private string no\_ordre;

private string op\_dissec;

#### Classe Present

public class Present

{

private int present\_id;

private Animal animalSivol;

private int troupeau\_id;

private DateTime d\_arrivee\_dom;

private DateTime d\_sortie\_dom;

#### Classe Animal

public class Animal

{

private int animal\_id;

private string espece;

private string dom\_nais;

private string annee\_nais;

private string lot\_nais;

private string no\_nais;

private string rfid;

private DateTime d\_nais;

private string lignee;

### private string sexe;

### Pour Ajouter/Modifier

En mode saisie : il faut Present\_id

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

{

string operation = btSaisir\_Valider.Text;

int iLigneCourrante = 0;

bool continuer = true;

string message = "";

Animal anim\_maj = new Animal();

anim\_maj.Dom\_Nais = txtSaisir\_Dom\_Nais.Text;

anim\_maj.Annee\_Nais = txtSaisir\_Annee\_Nais.Text;

anim\_maj.Lot\_Nais = txtSaisir\_Lot\_Nais.Text;

anim\_maj.No\_Nais = txtSaisir\_No\_Nais.Text;

anim\_maj.Sexe = txtSaisir\_Sexe.Text;

if (lblSaisir\_Animal\_Id.Text != "")

anim\_maj.Animal\_Id = int.Parse(lblSaisir\_Animal\_Id.Text);

Present pres\_maj = new Present();

pres\_maj.AnimalSivol = anim\_maj;

pres\_maj.Present\_Id = int.Parse(lblSaisir\_Present\_Id.Text);

Abattage abat\_add = new Abattage();

abat\_add.AnimalPresent = pres\_maj;

abat\_add.D\_Abattage = dtpSaisir\_D\_Abattage.Value;

abat\_add.Op\_Dissec = cbSaisir\_Op.Text;

switch (operation)

{

case "Ajouter":

continuer=AbattageDao.TableAbattage\_Ajouter\_UnEnregistrement(ora\_gen, abat\_add) ;

….etc…}

# Opérations sur les Cellules

Tables concernées Cellule\_Active , Cage

Il faut le cellule\_id et present\_id

# Opérations sur les Animaux en Cage

Ceci concerne les données liées à la partie Ponte et Repro :

Tables : PonteIndiv, PonteCollective, IncuIndiv, Plan\_Acc (cage\_id,cagemale\_id), Accouplement

*Il faudra donc récupérer ou trouver l’identifiant cage\_id*