I - INTRODUCTION

Comment gérer un planning sous Access ?...
Question souvent posée sur le forum. Je me propose donc au travers de la base de données "GESTION DE PLANNING" (téléchargeable) de démonter le mécanisme de création d'un planning.
Dans cette base, j'ai souhaité montrer un maximum d'éléments.
Quelques exemples :

- comment personnaliser les boutons et gérer les images (voir **Caféine**)
- comment appeler la palette des couleurs (Appeler une API - voir **Tofalu**)
- comment changer le pointeur de la souris (Appeler une API - voir **Faw**)
- comment enrichir le contenu d'une liste déroulante (voir **Maxence HUBICHE**)
- comment alimenter une liste déroulante en fonction d'une autre (voir **Jean-Philippe AMBROSINO**)
- manipuler les infos bulles (voir **Morgan BILLY**)
- listes déroulantes imbriquées, évènement sur déplacement de souris et autres, etc...

Cet article se divisera en deux parties.
1ère partie : Conception/structure de la base et Création du planning

Conception/structure de la base

1 Cahier des Charges
2 Structure des tables
3 Modèle relationnel

Création du planning (Formulaires nécessaires)

1 F_SaisieSalles (permet de saisir une nouvelle salle de formation)
2 F_SaisieCatalogue (permet de saisir le catalogue de formation)
3 F_SaisieFormateur (permet de saisir le profil d'un formateur)
4 F_Reservation (permet de créer une réservation)
5 F_Planning (Plate forme de toute l'application - affiche le planning et permet de générer une réservation, saisir une salle, un formateur, un nouvel item du catalogue)

2eme partie : Manipulations du planning, impression et envoi du planning des formateurs par mail

Manipulations du planning

1 Changer la date d'une formation
2 Changer de formateur
3 Changer de salle de formation
4 Supprimer une formation
5 Basculer l'affichage Salle de formation / Formateurs

Communiquer le planning

1 Imprimer le planning
2 Envoyer le planning du formateur par mail

*1) Nous ne traiterons pas de la gestion des barres de menus, vous trouverez toutes les informations nécessaires sur le tuto "Personnalisez vos barres de commandes dans Access" de **Starec**.*
2) Bien que le code VBA soit omniprésent dans l'application, je m'efforcerai d'aborder et d'expliquer les choses simplement afin de mettre cet article à la portée du plus grand nombre d'utilisateurs.

II - Conception et analyse

II-A - Le Cahier des Charges

Objet du projet :
Décrire le process de génération d'un planning sur Access.
Base de travail :
Suivre le planning de l'occupation des salles et des formateurs d'un centre de formation.
La saisie des formations se fait directement à partir du planning par clic sur une plage de celui-ci.
De manière à revenir sur l'imbrication des listes déroulantes et des liens entre listes (déroulantes ou non) avec sous formulaire, la saisie d'une formation se fera à partir de la famille du produit (liste déroulante) qui de ce fait affichera les produits concernés (Liste déroulante dépendante).
Les formations du Catalogue liées à la famille de produit et au produit s'afficheront dans un sous formulaire.
Par clic sur la formation choisie, on rafraichira la liste des formateurs compétents disponibles pour la formation.
Un résumé de la formation apparaitra dans une info-bulle sur le pavé correspondant du planning.
Dans l'idée de voir évoluer notre centre de formation, il sera possible d'ajouter de nouvelles salles. Cependant, les numéros devront rester séquentiels sans rupture de séquence.
D'autre part, une formation ne pourra être tronquée par un weekend.

II-B - Les Tables nécessaires

Je n'ai repris dans les tables que les champs strictement nécessaires à l'application

II-B-1 - T_Catalogue

Cette table stocke le descriptif des formations dispensées

Nom du Champ	Type de Données	Commentaires
CodeCatalogue	NuméroAuto	Clé Primaire
CodeProduit	Numérique	Entier long - En relation avec la table T_Produits sur le champ CodeProduit
CodeFamilleProduit	Numérique	Entier long - En relation avec la table T_FamilleProduit sur le champ CodeFamilleProduit
IntituleStage	Texte	255 car - Libellé de la formation
CodeNiveau	Numérique	Entier long - En relation avec la table T_NiveauFormation sur le champ CodeNiveau
DureeStage	Numérique	Octet - Correspond au nombre de jours du stage

II-B-2 - T_FamilleProduit

Cette table stocke les intitulés des différentes catégories de produit (Bureautique, Système,...)
Celle-ci sera enrichie à la volée.

Nom du Champ	Type de données	Commentaire
CodeFamilleProduit	NuméroAuto	Clé Primaire
NomFamilleProduit	Texte	255 car - Stocke l'intitulé de la catégorie de produit

II-B-3 - T_Formateur

Contient les infos administratives du formateur.

Nom du Champ	Type de données	Commentaire
CodeFormateur	NuméroAuto	Clé Primaire
Civilite	Texte	Représente les valeurs : Monsieur, Madame, Mademoiselle
NomFormateur	Texte	
PrenomFormateur	Texte	
Email	Texte	Permettra l'envoi du planning du formateur (sera traité en 2ième partie de l'article)

II-B-4 - T_NiveauFormation

Stocke les valeurs : Débutant, Avancé, Utilisateur... et sera enrichie à la volée

Nom du Champ	Type de données	Commentaire
CodeNiveau	NuméroAuto	Clé Primaire
LibelleNiveau	Texte	50 car - Intitulé du niveau de formation (Débutant ...)

II-B-5 - T_Planning

Cette table est le noeud du planning.
En effet, elle stocke autant d'enregistrements qu'il y a de jours de formation.
Ainsi, une formation de trois jours ajoutera 3 enregistrements dans la table.

Nom du Champ	Type de données	Commentaire
CodeStage	Numérique	Entier long - en relation avec la Clé Primaire de la table T_Stages
DatePlanning	Date/Heure	Jour de la formation
Quantième	Numérique	Octet - Se rapporte au n° d'ordre dans la formation (1er jour, 2ème jour ...). Sera utilisé dans l'info bulle commentant le pavé dans le formulaire du planning

II-B-6 - T_ProduitEnseigne

Cette table stocke le profil de formation des formateurs.
Par profil on comprendra : les produits enseignés ainsi que les niveaux de compétence.

Nom du Champ	Type de données	Commentaire
CodeProduitEnseigne	Numérique	Entier Long - en relation avec la table T_Produits par le champ CodeProduit
CodeNiveau	Numérique	Entier Long - en relation avec la table T_NiveauFormation par le champ CodeNiveau
CodeFormateur	Numérique	Entier Long - en relation avec la table T_Formateur par le champ CodeFormateur
CodeFamilleProduit	Numérique	Entier Long - en relation avec la table T_FamilleProduit par le champ CodeFamilleProduit

II-B-7 - T_Produits

Cette table décrit les produits qui sont enseignés dans le centre de formation.

Nom du Champ	Type de données	Commentaire
CodeProduit	NuméroAuto	Clé Primaire
NomProduit	Texte	50 Car - Libellé du produit
CodeFamilleProduit	Numérique	Entier long - en relation avec la table T_FamilleProduit sur le champ CodeFamilleProduit
CodeCouleur	Numérique	Entier long - Récupère le code couleur issu de la palette des couleurs

II-B-8 - T_SalleFormation

Cette table liste l'ensemble des salles de formation. Il est repris dans le Cahier des Charges que le numéro de salle sera sans rupture de séquence.
Pour respecter cette condition, nous choisirons un champ de type numérique que nous incrémenterons via le code.

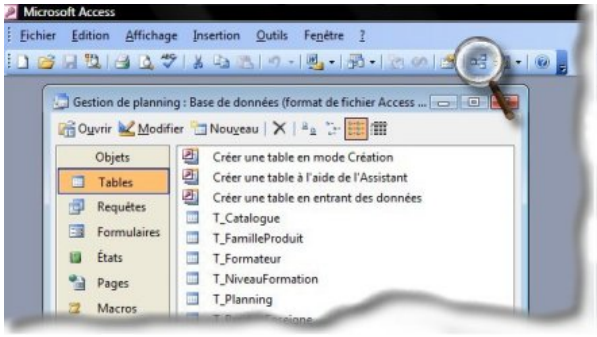
Nom du Champ	Type de données	Commentaires
CodeSalleFormation	Numérique	Clé Primaire - Taille : Octet - Sera incrémenté par calcul (contrainte du Cahier des Charges)
NomSalleFormation	Texte	Libellé du nom de la salle de formation

II-B-9 - T_Stages

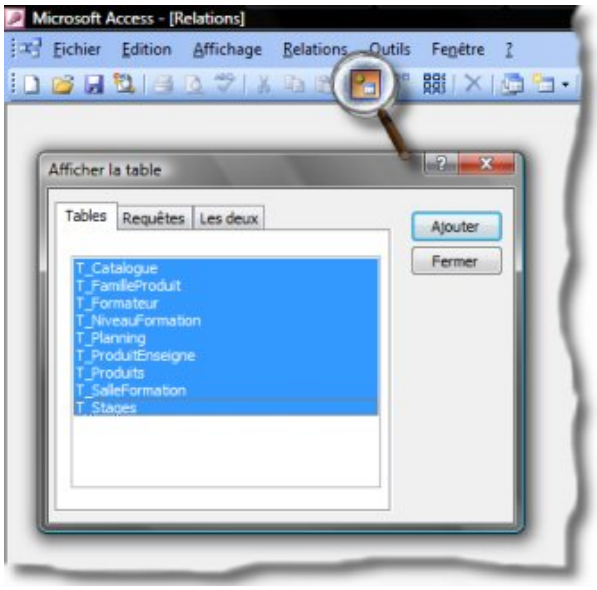
Cette table stocke le descriptif d'une réservation.

Nom du Champ	Type de données	Commentaire
CodeStage	NuméroAuto	Clé Primaire
DateStageDebut	Date/Heure	Date récupérée en cliquant sur le planning lors de la réservation
CodeSalleFormation	Numérique	Octet - en relation avec la table T_SalleFormation sur le champ CodeSalleFormation. Code récupéré en cliquant sur le planning lors de la réservation
CodeCatalogue	Numérique	Entier long - en relation avec la table T_Catalogue sur le champ CodeCatalogue
CodeFormateur	Numérique	Entier long - en relation avec la table T_Formateur sur le champ CodeFormateur

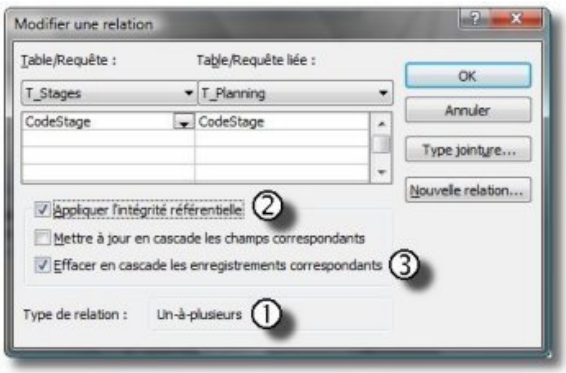
II-C - Le Modèle relationnel



Une fois les tables établies, il est maintenant temps de mettre en place les relations entre celles-ci.
Pour mettre en place les relations, il faut ouvrir la fenêtre des relations en cliquant sur l'outil "Relations" dans la barre d'outils de la fenêtre "Base de données".



Cliquez sur l'outil "Ajouter une table" et choisir dans la boite de dialogue les tables concernées par l'opération.



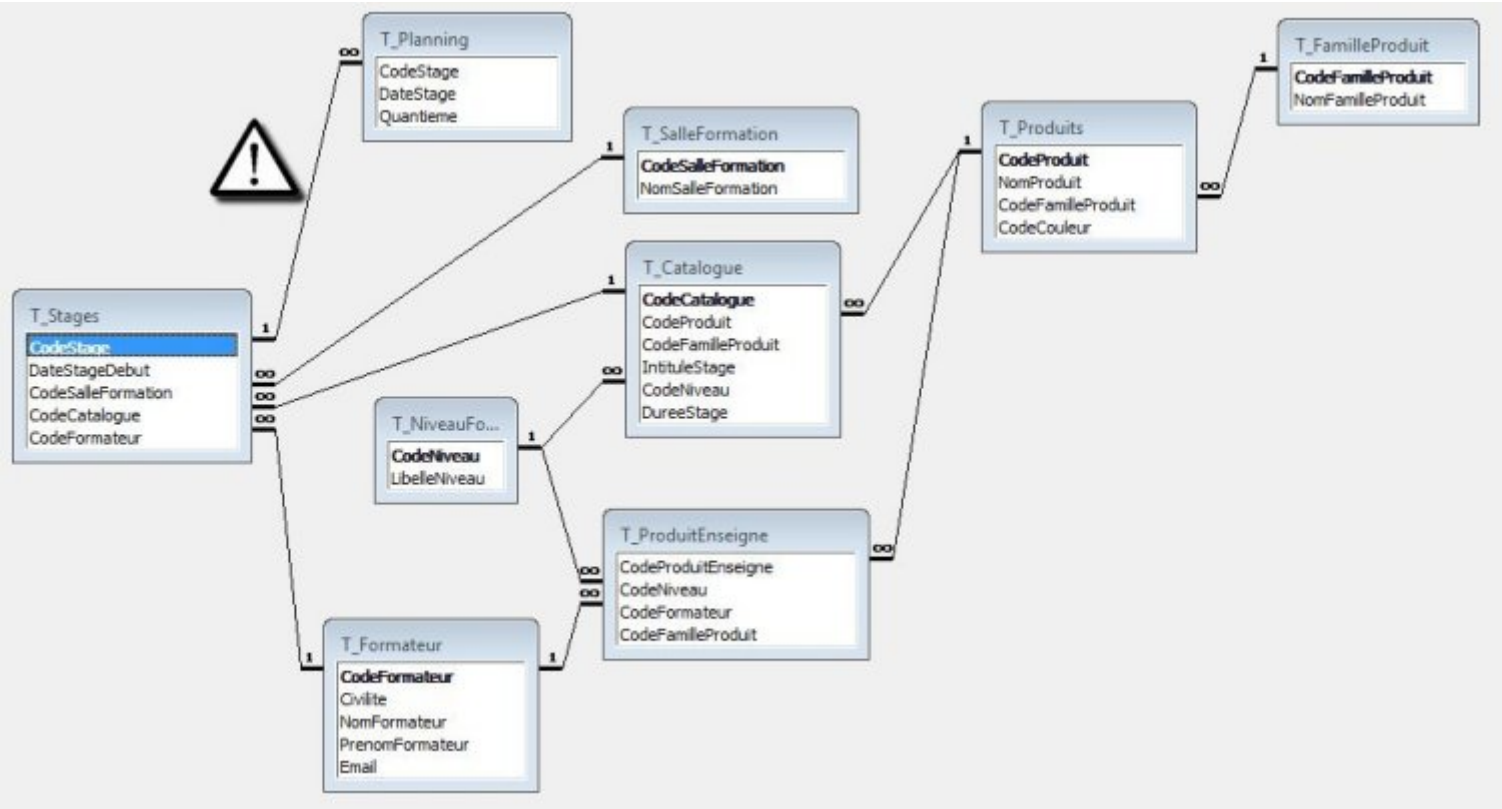
Pour créer une relation, il suffit de cliquer maintenu sur le champ de la table MERE et de glisser sur le champ à mettre en relation dans la table FILLE.
En lâchant la souris, Access affiche la boite de dialogue ci-contre.

***1** - Access affiche le type de relation.*

***2** - Cocher cette option pour activer l'intégrité référentielle. Cette option a pour effet de contrôler la cohérence des données. Cela signifie :*
qu'il sera impossible de supprimer une enregistrement PERE si des enregistrements FILS sont attachés.
***exemple** : Je ne pourrai pas supprimer un formateur si des stages lui sont attribués.*
De même, je ne pourrai pas créer un enregistrement FILS si l'enregistrement PERE, de la table liée, n'existe pas.
***exemple** : Je ne pourrai pas créer un nouveau produit si la famille de ce produit n'existe pas.*

3 *- En activant cette option, Access supprimera tous les enregistrements des tables filles attachés à l'enregistrement PERE supprimé.*
exemple *: Nous activerons cette option sur la relation qui existe entre T_Stages et T_Planning. Ainsi la suppression d'un stage entrainera la suppression de toutes les lignes de la table T_Planning attachées à ce stage.*

Lorsque toutes les relations ont été créées, nous obtenons l'écran suivant



III - Les Formulaires

III-A - Principes des boutons personnalisés

Il est possible de créer ses propres boutons et de leur donner un aspect dynamique.
Cette section va présenter ce processus. Le même processus est utilisé sur tous les boutons dessinés dans les différents formulaires

III-A-1 - Description des boutons

Pour pouvoir donner l'effet d'animation, 3 images seront nécessaires.

N°	Nom du bouton	Commentaire
1	btnFermerR	**L'effet relâché**. Visible par défaut, sera la première image de la pile
2	btnFermerS	**L'effet survolé**. Placée sous l'image 1, sera visible lors du passage de la souris sur le "bouton relaché"
3	btnFermerC	**L'effet cliqué**. Placée sous l'image 2, sera rendu visible lorsque l'on appuie sur le bouton de la souris.

III-A-2 - Mécanisme de l'animation

Pour donner l'illusion d'un clic, il faut décomposer le mouvement :

1 Je survole le bouton qui est au premier plan.
Celui-ci doit donc s'effacer pour montrer le bouton effet survolé. Cet effet permet de donner l'illusion à l'utilisateur que le bouton est sélectionné.
On agira donc sur l'évènement "sur souris déplacée"

2 Je clique sur le bouton mis en évidence.
Celui-ci doit donc s'effacer pour afficher le bouton effet cliqué.
On agira donc sur l'évènement "sur souris appuyée".

3 Je lâche la souris. Le geste du clic sera donc complet. En relâchant la souris, le bouton effet cliqué doit s'effacer pour remontrer l'effet survolé puisque ma souris est toujours au dessus du bouton.
On agira donc sur l'évènement "sur souris relâchée".

Cependant, lorsque la souris ne survole pas de bouton, il faut que l'effet relâché revienne au premier plan. Pour donner cet effet, on agira sur l'évènement "souris relâchée" de la section du formulaire où se trouve le bouton.

III-A-3 - Le Code VBA

III-A-3-a - Chargement des images

Les images sont toutes stockées dans un dossier "image". Celles-ci sont chargées à l'ouverture de chaque formulaire.
Exemple de chargement des images du formulaire : F_SaisieSalle

```
Private Sub Form_Open(Cancel As Integer)
    ' Chargement des images
    btnFermerR.Picture = CurrentProject.Path & "\image\btnFermerR.jpg"
```

```
    btnFermerS.Picture = CurrentProject.Path & "\image\btnFermerS.jpg"
    btnFermerC.Picture = CurrentProject.Path & "\image\btnFermerC.jpg"
    btnNouveauR.Picture = CurrentProject.Path & "\image\btnNouveauR.jpg"
    btnNouveauS.Picture = CurrentProject.Path & "\image\btnNouveauS.jpg"
    btnNouveauC.Picture = CurrentProject.Path & "\image\btnNouveauC.jpg"
End Sub
```

*Pour plus d'informations sur la gestion d'images avec Access, lire le **tuto de Caféine***

Ce code sera placé sur l'évènement "sur ouverture" du formulaire.
Le chemin de stockage de l'image est donné par la concaténation du chemin de l'application et du nom du fichier image à récupérer.
Ce code est bien sûr fonction du nombre de boutons à charger dans le formulaire.

III-A-3-b - Le code VBA pour les évènements de type "Déplacé" et "Appuyé"

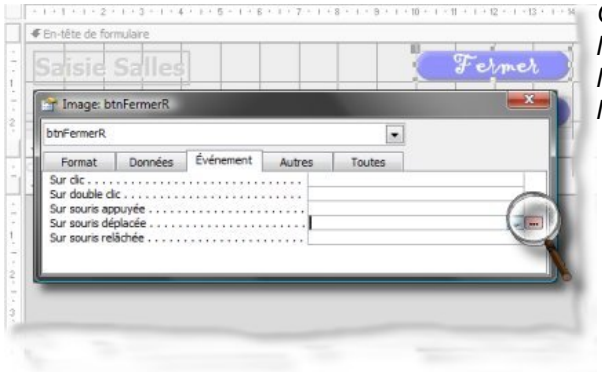
Pour les évènements "sur souris déplacée" et "sur souris appuyée", on utilisera une fonction qui réagira en rapport avec les images à afficher ou à masquer.
Descriptif de la fonction :

```
Function BasculerBouton(CtlMasque As String, CtlAffiche As String)
' frmActif représente le formulaire en cours d'utilisation
    frmActif.Controls(CtlAffiche).Visible = True
    frmActif.Controls(CtlMasque).Visible = False
End Function
```

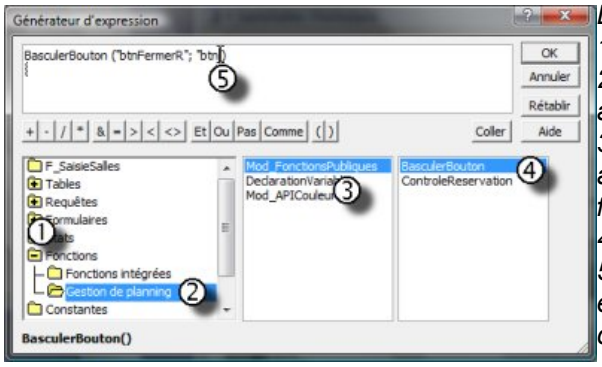
Au niveau des arguments, on indiquera le nom des boutons concernés par l'opération de bascule.
On remarque dans le code : "**frmActif**".
Il s'agit là d'une variable publique déclarée, dans un module spécifique, par :

```
' Variable pour récupération du formulaire en cours d'utilisation pour animation des boutons
Public frmActif As Form
```

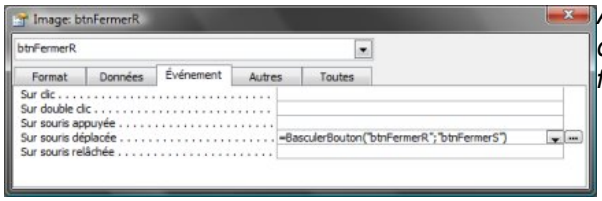
L'implantation de la fonction sur les boutons se fera via la fenêtre des Propriétés.



Ouvrir la fenêtre des propriétés. Dans l'onglet "Evènements", cliquez dans la ligne "sur souris déplacée" puis cliquez sur le bouton affiché dans la loupe (rosé).



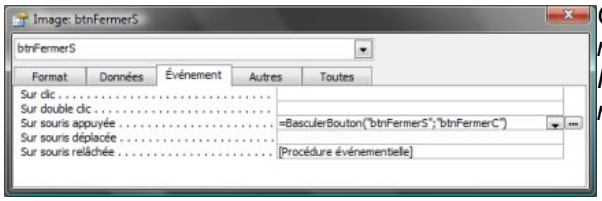
Description de la procédure :
1) Double clic sur le dossier "Fonctions".
2) Un Clic sur "Gestion Planning" pour avoir accès aux fonctions de l'application.
3) Un Clic sur "mod_FonctionsPubliques" afin d'afficher dans le volet de droite les fonctions disponibles.
4) Double Clic sur la fonction à utiliser.
5) Compléter les arguments de la fonction en saisissant les noms des contrôles concernés.



Après validation de la boite de dialogue du générateur d'expression, on obtient la formule ci-contre.

Ainsi sur les boutons on aura :
- type Relaché : =BasculerBouton("btnFermerR";"btnFermerS").
- type Survolé : =BasculerBouton("btnFermerS";"btnFermerC").

III-A-3-c - Le code VBA pour l'évènement "Relâché"



Ce code est placé sur l'évènement "souris relâchée" du bouton de type "S".
Il sera bien sûr fonction des opérations à réaliser.

Au plus simple nous pourrions avoir :

```
Private Sub btnFermerS_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single)
    ' Appel de la routine de bascule des images
    Call BasculerBouton("btnFermerC", "btnFermerS")
 ' Ferme le formulaire en cours
    DoCmd.Close
End Sub
```

Le code de chaque évènement "sur souris relâchée" sera étudié avec le formulaire correspondant.

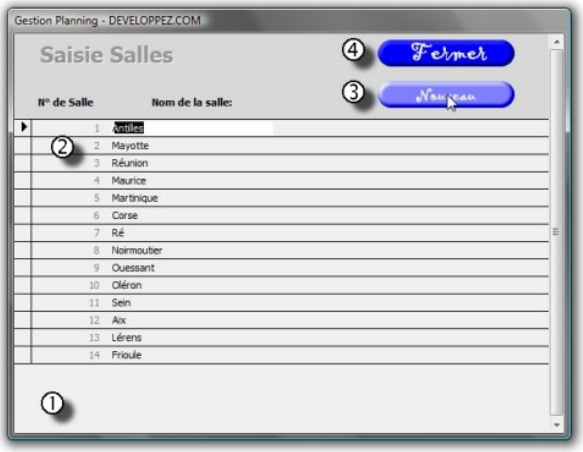
III-A-3-d - Le Code VBA pour la section de formulaire survolé

```
Private Sub EntêteFormulaire_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)
    ' intitialisation de la variable objet public frmActif
    Set frmActif = Me

    ' Initialisation de l'affichage des boutons
    btnFermerR.Visible = True
    btnFermerS.Visible = False
    btnFermerC.Visible = False
    btnNouveauR.Visible = True
```

```
    btnNouveauS.Visible = False
    btnNouveauC.Visible = False

End Sub
```

Ce code sera bien sûr fonction du nombre de boutons affichés dans la section du formulaire

III-B - Le formulaire : F_SaisieSalles

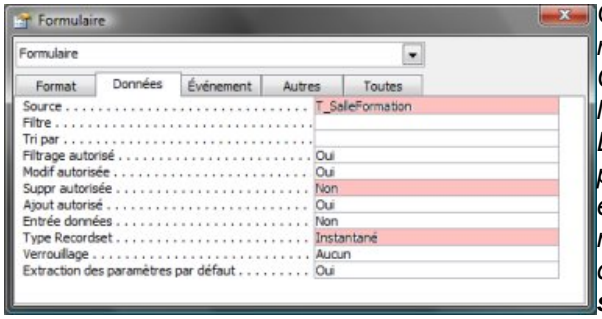


Lors de la création d'un formulaire, il nous faut définir l'objectif de celui-ci.
Par la méthode du TOE (Tâche - Objet - Evènement) on précisera les objets qu'il contiendra, les tâches que ceux-ci auront à accomplir et à quel moment (sur quel évènement).
***L'objectif :** :*
Ce formulaire permettra à l'utilisateur de saisir de nouvelles salles. Cependant certaines contraintes ont été posées par le Cahier des Charges. (Pour mémoire : Numéro en continu sans rupture de séquence).
Pour réaliser cela, nous sécuriserons le formulaire et le numéro de la salle sera obtenu via un calcul.
Les tâches :
Créer un nouveau n° de salle.
Vérifier que l'enregistrement est complet.
Les objets :
Le contrôle "CodeSalleFormation" ne sera pas accessible par l'utilisateur. L'incrémentation se faisant par le code.
Le contrôle "NomSalleFormation", contrôle dépendant de la source du formulaire, permettra de compléter l'enregistrement.
Le bouton "Fermer" dont la tâche sera de vérifier que l'enregistrement est complet.
Le bouton "Nouveau" dont la tâche sera de donner accès à une nouvelle saisie et de générer le n° de la salle.
Les évènements :
*Le CodeSalle, étant inaccessible, sera généré **"sur entrée"** dans le contrôle "NomSalleFormation".*
*La vérification de l'enregistrement complet se fera **"sur souris relâchée"** du bouton "btnFermerS"*
*L'accès à une nouvelle saisie se fera **"sur souris relâchée"** du bouton "btnNouveauS".*

III-B-1 - Propriétés du formulaire

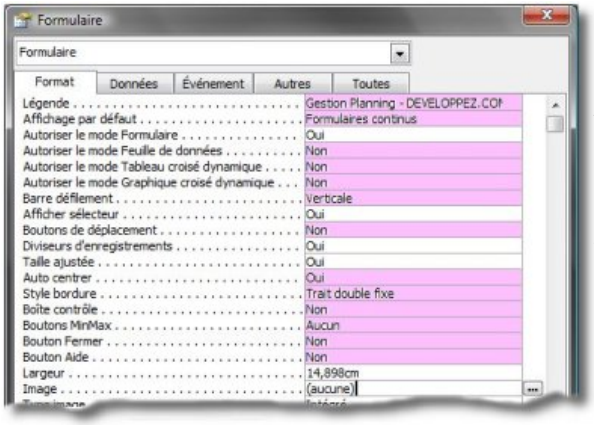
Le formulaire sera basé sur la table T_SalleFormation (dont les caractéristiques ont été définies plus haut).
Afin d'avoir une vue d'ensemble des propriétés modifiées, celles-ci sont colorées.

III-B-1-a - Propriétés : Données



On remarquera la propriété : Type recordset (1 sur l'image).
Celle-ci positionnée sur "Instantané" rend les données du formulaire inaccessibles.
*D'autre part, en ayant positionné la propriété : "suppr autorisée" sur **Non**, nous évitons les accidents de suppression et nous respectons la contrainte du Cahier des Charges à savoir : **pas de rupture de séquence**.*

III-B-1-b - Propriétés : Format

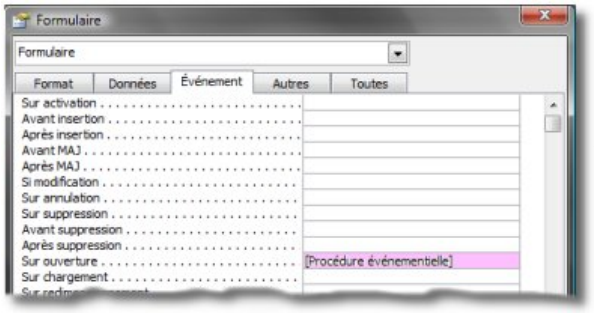


Beaucoup de propriétés ont été modifiées.
Tout d'abord le verrouillage du type de formulaire par l'affichage par défaut et par l'autorisation ou non d'autres types.
Ensuite dans la barre de titre du formulaire (2 sur l'image) où l'intitulé a été personnalisé par légende, tous les boutons sont supprimés (dont le bouton "Fermer"). Ceci évitera de fermer sur un enregistrement sans nom de salle.

III-B-1-c - Propriétés : Autres

Deux propriétés modifiées afin de garder le formulaire en premier plan tant que celui-ci ne sera pas fermé via le bouton personnalisé "FERMER" du formulaire.

III-B-1-d - Propriétés : Evènements

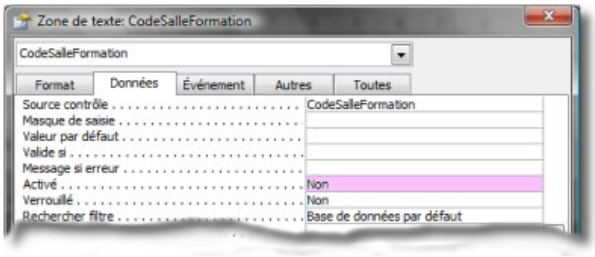


Une seule propriété modifiée : Sur Ouverture. Il s'agit en fait de la procédure de chargement des images.
*Comme décrit dans la section : **Evènement "sur Ouverture" du formulaire**.*

III-B-2 - Le contrôle : CodeSalleFormation

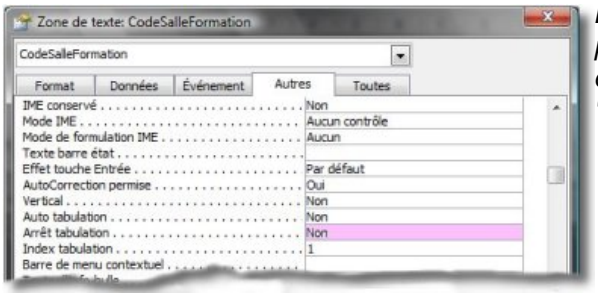
Hormis les boutons, déjà présentés plus haut, j'attire l'attention sur le contrôle "CodeSalleFormation".
En effet, celui-ci sera généré par le code. De ce fait, des propriétés lui seront affectées en conséquence.

III-B-2-a - Propriétés : Données



*La propriété "Activé" a été paramétrée sur **NON**. Ainsi, l'utilisateur n'y a pas accès, il apparaîtra grisé sur le formulaire*

III-B-2-b - Propriétés : Autres



*La propriété "Arrêt Tabulation" a été paramétrée sur **NON**. Ainsi le focus entre directement dans le contrôle "NomSalleFormation".*

III-B-3 - Les évènements

Je vais maintenant détailler les différentes procédures gérées par les évènements du formulaire et des contrôles

III-B-3-a - Evènement "sur Ouverture" du formulaire

```
Private Sub Form_Open(Cancel As Integer)
    ' Chargement des images
    btnFermerR.Picture = CurrentProject.Path & "\image\btnFermerR.jpg"
    btnFermerS.Picture = CurrentProject.Path & "\image\btnFermerS.jpg"
    btnFermerC.Picture = CurrentProject.Path & "\image\btnFermerC.jpg"
    btnNouveauR.Picture = CurrentProject.Path & "\image\btnNouveauR.jpg"
    btnNouveauS.Picture = CurrentProject.Path & "\image\btnNouveauS.jpg"
    btnNouveauC.Picture = CurrentProject.Path & "\image\btnNouveauC.jpg"
    ' Variable qui récupère le premier CodeSalleFormation
    intCodeSalle = 1
End Sub
```

Ce code appelle une petite remarque :
La variable **intCodeSalle** est une variable publique déclarée dans le module de **DéclarationsVariables**.
Son objet : permettra de savoir si une nouvelle salle a été saisie.
voir le code sur le bouton FERMER ci-dessous)

III-B-3-b - Evènement "sur Entrée" du contrôle NomSalleFormation

```
Private Sub NomSalleFormation_Enter()
 ' Test pour connaître la position de notre focus : Nouvel Enregistrement ou Premier enregistrement du Formulaire
    If CodeSalleFormation = 0 Then
        'Incrémente le dernier n° de salle
        Me.CodeSalleFormation = DMax("CodeSalleFormation", "T_SalleFormation") + 1
    End If
End Sub
```

Pour compléter votre connaissance sur les "Fonctions de Domaine sous Access", je vous engage à consulter le tuto de Starec

III-B-3-c - Evènement "sur Souris Relâchée" du bouton btnNouveauS

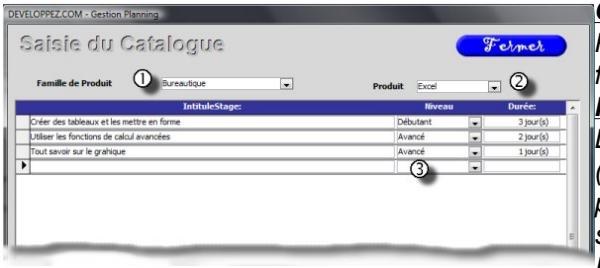
```
Private Sub btnNouveauS_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single)
    ' Appel de la procédure de bascule des images
    Call BasculerBouton("btnNouveauC", "btnNouveauS")

    ' Réactive le type Recordset en dynamique
    Me.RecordsetType = 0
    ' Autorise l'entrée de nouvelles données
    Me.AllowAdditions = True

    'Positionne le curseur sur le nouvel enregistrement
    DoCmd.GoToRecord acDataForm, "F_SaisieSalles", acNewRec
    ' intitialisation de la variable de contrôle de saisie d'un nouvel enregistrement
    intCodeSalle = 0
End Sub
```

III-B-3-d - Evènement "sur Souris Relâchée" du bouton btnFermerS

```
Private Sub btnFermerS_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single)
Dim strSqlSupprNouvelEnregistrement As String

' Structure de la requête de suppression de l'enregistrement incomplet
strSqlSupprNouvelEnregistrement = "DELETE NomSalleFormation FROM T_SalleFormation WHERE NomSalleFormation Is Null

    ' Appel de la routine de bascule des images
    Call BasculerBouton("btnFermerC", "btnFermerS")

    If intCodeSalle > 0 Then
        ' Fermeture du formulaire, aucune saisie n'est lancée
        DoCmd.Close
    Else
        ' Affiche la boite de dialogue d'alerte - un nouvel enregistrement est en cours
        intReponse = MsgBox("La saisie de ce champ est obligatoire" & vbCrLf & _
      "Souhaitez-vous interrompre l'opération ?", vbQuestion + vbYesNo, cstDVP)

        ' Teste la réponse de l'utilisateur
        If intReponse = vbYes Then
            ' Désactivation des messages système
            DoCmd.SetWarnings False
            ' Sauvegarde l'enregistrement incomplet
            DoCmd.RunCommand acCmdSaveRecord
            'Supprime l'enregistrement incomplet
            DoCmd.RunSQL strSqlSupprNouvelEnregistrement
            ' Réactivation des messages système
            DoCmd.SetWarnings True
            ' Ferme le formulaire
            DoCmd.Close
        End If
    End If
```

```
End Sub
```

III-C - Le formulaire : F_Catalogue

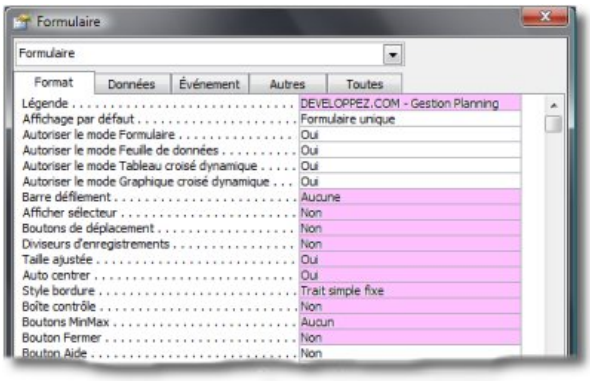


Objectif du formulaire *:*
Permettre la création du Catalogue de formations dispensées.
Fonctionnement *:*
L'utilisateur choisit une famille de produits (1) (Bureautique, Système, Langage, ...) puis dans la liste des produits (2), sélectionne le produit approprié.
Il pourra alors saisir en série les intitulés des stages mis à disposition en précisant leur niveau (3) et leur durée.
Les tâches :
La liste déroulante des produits est influencée par la liste des familles de produits.
Chaque produit est associé à une couleur que l'on déterminera à la création du produit.
les objets :
Le formulaire en lui-même est un formulaire indépendant sur lequel seront déposés les deux listes déroulantes et un sous-formulaire.
La liste cboFamilleProduit : contrôle indépendant dont la source est une requête basée sur la table T_FamilleProduit.
La liste cboProduit : contrôle indépendant dont la source est une requête basée sur la table T_Produits mais dont le champ codeFamilleProduit est dépendant de la liste cboFamilleProduit.
Le sous formulaire SF_SaisieCatalogue : contrôle dépendant de la table T_Catalogue rattaché aux deux listes cboFamilleProduit et cboProduit.
Les évènements :
Les trois listes seront mises à jour à la volée. Cela signifie donc que l'on agira sur l'évènement "sur Absence dans liste" de chacune de celles-ci.

III-C-1 - Les propriétés du formulaire

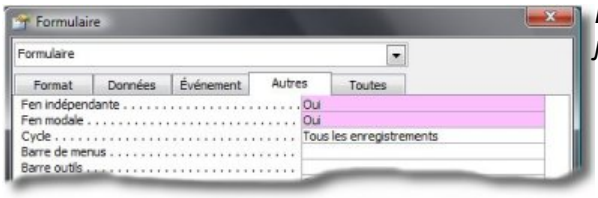
Le formulaire est indépendant donc pas de source de données.
Repérons les différentes propriétés de celui-ci

III-C-1-a - Propriétés : Format



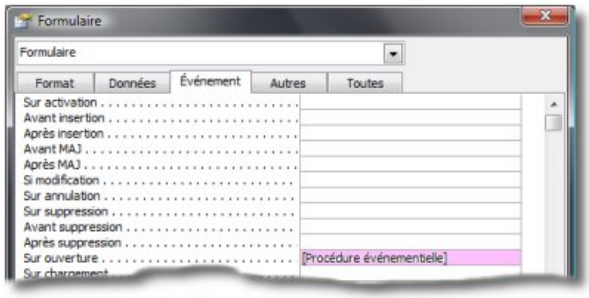
Comme pour le formulaire précédent, j'ai supprimé l'ensemble des boutons puisque je gère la fermeture du formulaire avec le bouton : btnFermerS.
A noter également que j'ai personnalisé l'intitulé du formulaire en modifiant la propriété "Légende".

III-C-1-b - Propriétés : Autres



Le formulaire restera donc au premier plan jusqu'à fermeture de celui-ci.

III-C-1-c - Propriétés : Evènements



Comme pour les autres formulaires, on retrouvera le chargement des images de boutons.

```
Private Sub Form_Open(Cancel As Integer)
    ' Charge les images
    Me.btnFermerR.Picture = CurrentProject.Path & "\Image\btnFermerR.jpg"
    Me.btnFermerS.Picture = CurrentProject.Path & "\Image\btnFermerS.jpg"
    Me.btnFermerC.Picture = CurrentProject.Path & "\Image\btnFermerC.jpg"
End Sub
```

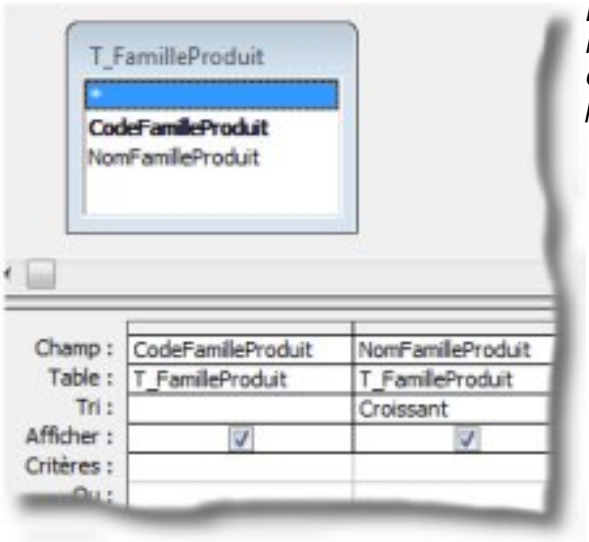
En relation avec le chargement des images, on aura forcément la gestion du survol du bouton "FERMER".
On aura donc une procédure sur l'évènement "sur Souris Déplacée" de l'entête de formulaire.

```
Private Sub EntêteFormulaire_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)
    ' récupère le formulaire en cours
    Set frmActif = Me
    btnFermerR.Visible = True
    btnFermerS.Visible = False
    btnFermerC.Visible = False
End Sub
```

III-C-2 - Le contrôle cboFamilleProduit

Liste déroulante permettant de choisir une famille de produit. Cette liste sera mise à jour à la volée.

III-C-2-a - Propriétés : Données



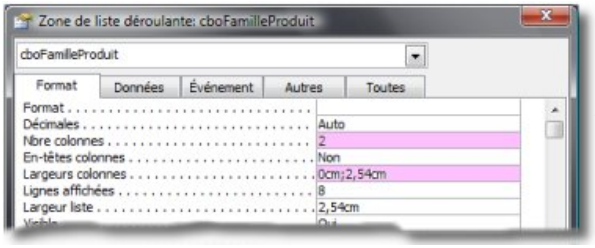
Lors de la création du contrôle, l'assistant nous a permis de construire la requête ci-contre. Ce qui se traduit en langage SQL par :

```
SELECT T_FamilleProduit.CodeFamilleProduit, T_FamilleProduit.NomFamilleProduit
FROM T_FamilleProduit
ORDER BY [NomFamilleProduit];
```

Les propriétés résultantes sont colorées en rose.

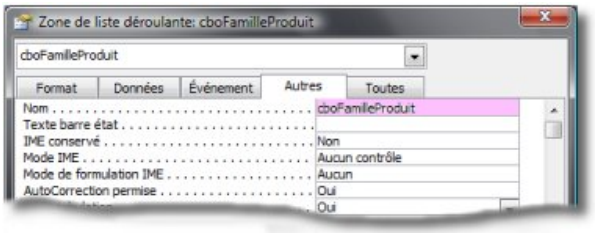
On remarquera :
la propriété : Limiter à la liste = **OUI**. Celle-ci permettra de gérer l'évènement "sur Absence dans liste".
la propriété : Colonne liée = 1. Cette colonne contient le codeFamilleProduit. Il sera mémorisé dans le formulaire et servira de premier champ PERE pour la liaison avec le sous-formulaire.

III-C-2-b - Propriétés : Format



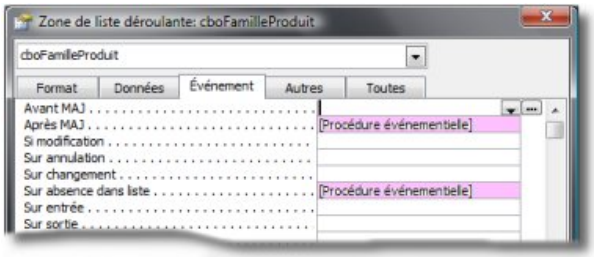
On remarquera :
la propriété : Nombre de colonnes = 2. Correspond au nombre de champs dans la requête.
la propriété : Largeur de colonnes = 0cm;2,54cm. La largeur 0 correspond à une colonne masquée.

III-C-2-c - Propriétés : Autres



Cet onglet permet d'identifier le contrôle. Il est préfixé de "cbo" pour désigner une combobox (Liste déroulante).

III-C-2-d - Propriétés : Evènements



On remarquera que deux évènements sont contrôlés.

L'évènement : **Après Mise à jour**.
Il permet de mettre à jour et d'atteindre le contrôle cboProduit

```
Private Sub cboFamilleProduit_AfterUpdate()
    ' réactualise la liste des produits en fonction de la famille choisie
    Me.cboProduit.Requery
    ' Envoie le focus dans le contrôle cboProduit
    ' Objectif : Développer automatiquement le contrôle : cboProduit
    ' En conséquence, l'évènement "sur Entrée" du contrôle cboProduit sera donc géré.
    DoCmd.GoToControl "cboProduit"
End Sub
```

L'évènement : **Sur Absence dans liste**.
Permet d'ajouter à la volée une nouvelle valeur dans la liste.

```
Private Sub cboFamilleProduit_NotInList(NewData As String, Response As Integer)
    ' Appel de la routine publique AjoutDansListe
    Call AjoutDansListe("T_FamilleProduit", "NomFamilleProduit", NewData)
    ' Rafraîchit la liste "cboFamilleProduit"
    Response = acDataErrAdded
End Sub
```

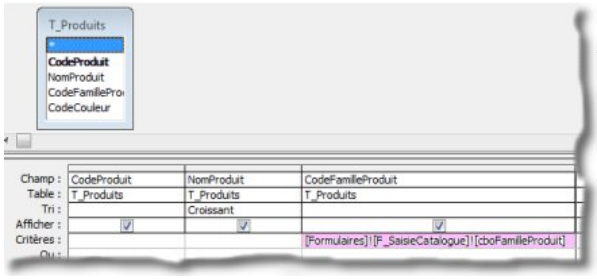
Voici le détail de la routine AjoutDansListe :

```
Public Sub AjoutDansListe(strTable As String, strChamp As String, NouvelleValeur As String)
    ' Désactivation des messages système
    DoCmd.SetWarnings False
    ' Exécution de la requête ajout afin d'inclure la nouvelle saisie dans la liste
    ' Code proposé par Maxence HUBICHE dans la F.A.Q ACCESS
    DoCmd.RunSQL "INSERT INTO " & strTable & "( " & strChamp & " ) SELECT """ & NouvelleValeur & """;"
    ' Réactivation des messages système
    DoCmd.SetWarnings True
End Sub
```

III-C-3 - Le contrôle : cboProduit

Liste déroulante liée à cboFamilleProduit. Cette liste sera enrichie à la volée. Une couleur sera associée à chaque produit saisi par appel de l'API ouvrant la palette de couleurs de Windows.

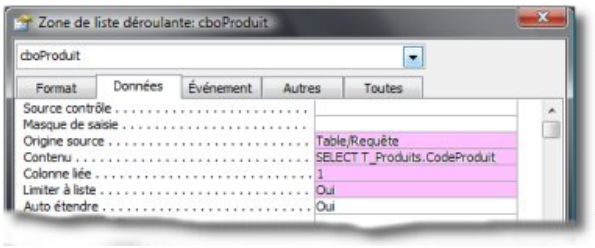
III-C-3-a - Propriétés : Données



On remarquera le troisième champ de la requête dont le critère fait appel au contrôle cboFamilleProduit du formulaire. Nous aurons un lien entre les deux listes. Pour plus d'informations sur la gestion des listes, ***consulter la F.A.Q d'Access***

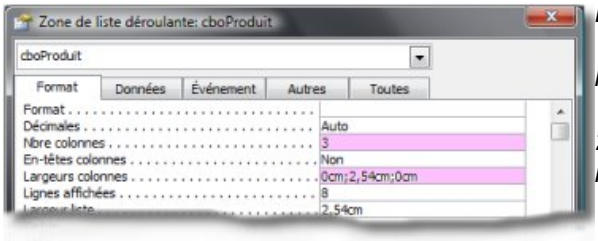
Nous obtenons le code SQL suivant :

```
SELECT T_Produits.CodeProduit, T_Produits.NomProduit, T_Produits.CodeFamilleProduit
FROM T_Produits
WHERE T_Produits.CodeFamilleProduit=[Formulaires]![F_SaisieCatalogue]![cboFamilleProduit]
ORDER BY T_Produits.NomProduit;
```



On constatera que les propriétés modifiées sont les mêmes que pour le contrôle cboFamilleProduit. La colonne liée est la colonne contenant le CodeProduit et la valeur sera mémorisée dans le formulaire. Elle servira de second champ PERE pour la liaison avec le sous-formulaire.

III-C-3-b - Propriétés : Format

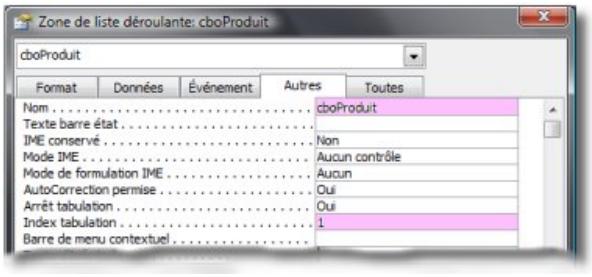


Deux propriétés ont été modifiées :

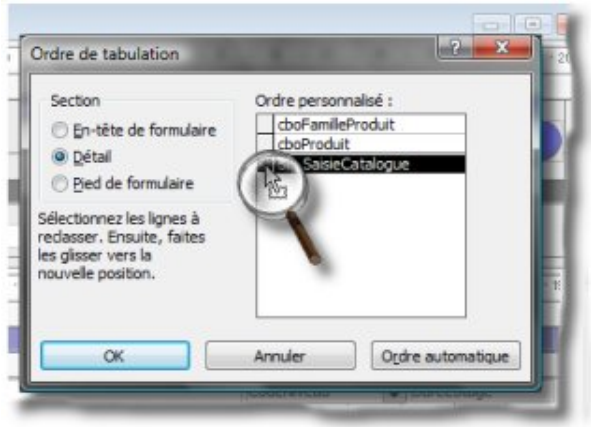
- Nombre de colonnes = 3. Représente le nombre de colonnes dans la requête.

- Largeur des colonnes = 0cm; 2,54cm; 0cm. Deux colonnes sont à 0 et n'apparaîtront pas dans la liste.

III-C-3-c - Propriétés : Autres

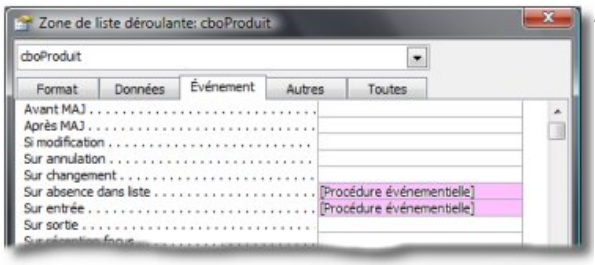


Deux propriétés ont été modifiées :
- Nom = cboProduit. Je garde la même structure que pour la liste précédente.
- Index Tabulation = 1. Cet index indique à Access l'ordre dans lequel le focus doit se déplacer dans le formulaire.



Pour modifier l'ordre des tabulations :
1) Cliquez droit dans le formulaire en mode création.
2) Cliquez "Ordre de tabulation".
3) Cliquez maintenu sur le sélecteur correspondant au contrôle à déplacer et glissez.

III-C-3-d - Propriétés : Evènements



Deux évènements sont contrôlés :

L'évènement **"sur Entrée"**.
La procédure a pour objet de développer la liste dès que le focus est entré dans le contrôle.
La procédure aurait pu être placée également sur l'évènement "sur Réception focus".

```
Private Sub cboProduit_Enter()
    ' Ouvre la liste déroulante dès réception du focus
    cboProduit.Dropdown
End Sub
```

L'évènement **"sur Absence dans liste"** qui permettra l'ajout d'une nouvelle valeur.
On remarquera dans la procédure l'utilisation d'un recordset pour gérer l'enregistrement ajouté.
Pour plus d'informations sur D.A.O, rendez-vous sur le tuto : **Créer et Manipuler des données avec D.A.O**

```
Private Sub cboProduit_NotInList(NewData As String, Response As Integer)
    'Déclaration des variables
    Dim rsProduits As DAO.Recordset
```

```
    Dim strSqlProduits As String
    Dim lngCodeCouleur As Long

    ' Inititialistion des variables
    strSqlProduits = "SELECT * FROM T_Produits"
    Set rsProduits = CurrentDb.OpenRecordset(strSqlProduits)

    ' avertissement à l'utilisateur
    MsgBox "La palette de couleurs va s'afficher afin d'affecter une couleur au nouveau produit." _
 & vcrlf & "Cette couleur sera reprise sur le planning.", vbInformation, cstDVP

 ' ****************************************************
 ' appel de la palette de couleurs - Utilisation d'une API
    ' Code proposé par Tofalu
 ' ****************************************************

' Définition pour relancer la définition d'une couleur
Relance:
    lngCodeCouleur = ShowColor(Me.hwnd)

    ' Contrôle que la couleur choisie ne correspond pas à la couleur weekend ou à la couleur par défaut
    If lngCodeCouleur = 8454143 Or lngCodeCouleur = 16777215 Then
        MsgBox "la couleur choisie est une couleur réservée pour l'application", vbInformation, cstDVP
        ' renvoie sur l'étiquette et réaffiche la boite de couleurs
        GoTo Relance
    End If

    ' ****************************************************

    With rsProduits
        ' ajout d'un enregistrement
        .AddNew
            ' renseignement des champs
                ' Ajoute le nouveau produit
                .Fields(1) = NewData
                ' Stocke la famille du produit
                .Fields(2) = cboFamilleProduit
                ' Stocke la nouvelle couleur
                .Fields(3) = lngCodeCouleur
        ' Mise à jour de la table
        .Update
    End With

    ' Mise à jour de la liste
    Response = acDataErrAdded
End Sub
```

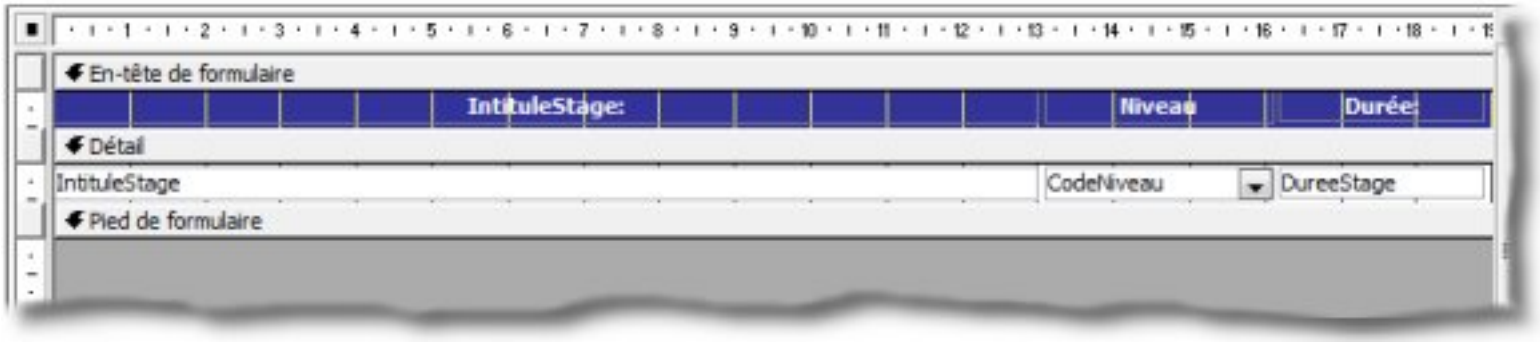
On peut remarquer dans le code la variable lngCodeCouleur intialisée par **ShowColor(Me.Hwnd)**. Cette instruction correspond à l'appel d'une A.P.I (Application Programming Interface).
Une A.P.I est un sous programme stocké dans une bibliothèque (Fichier DLL). J'ai récupéré **ce code dans les sources de DVP** proposé par Tofalu .

```
Private Declare Function CHOOSECOLOR Lib "comdlg32.dll" Alias _
"ChooseColorA" (pChoosecolor As CHOOSECOLOR) As Long
Private Type CHOOSECOLOR
    lStructSize As Long
    hwndOwner As Long
    hInstance As Long
    rgbResult As Long
    lpCustColors As String
    flags As Long
    lCustData As Long
    lpfnHook As Long
    lpTemplateName As String
End Type

Public Function ShowColor(Handle As Long) As Long
    Dim cc As CHOOSECOLOR
    Dim Custcolor(16) As Long
```

```
    Dim lReturn As Long
    Dim CustomColors

    'set the structure size
    cc.lStructSize = Len(cc)
    'Set the owner
    cc.hwndOwner = Handle
    'set the custom colors (converted to Unicode)
    cc.lpCustColors = StrConv(CustomColors, vbUnicode)
    'no extra flags
    cc.flags = 0

    'Show the 'Select Color'-dialog
    If CHOOSECOLOR(cc) <> 0 Then
        ShowColor = cc.rgbResult
        CustomColors = StrConv(cc.lpCustColors, vbFromUnicode)
    Else
        ShowColor = -1
    End If
End Function
```

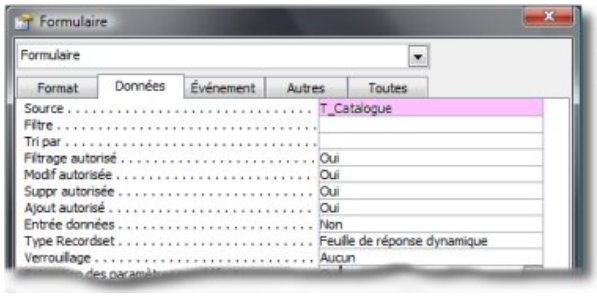
Ce code sera implanté dans un module standard. (Voir le module : Mod_APICouleur de la base exemple)

III-C-4 - Le contrôle Sous Formulaire



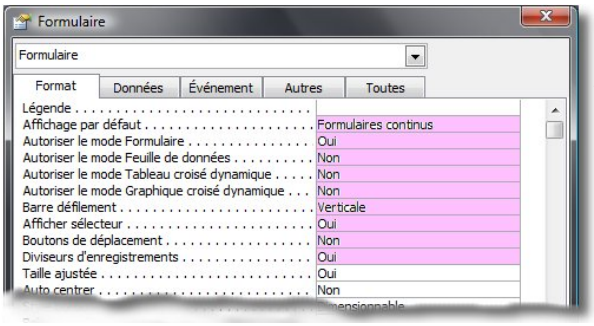
Ce formulaire est attaché à la table T_Catalogue. Il est relié aux deux listes déroulantes cboFamilleProduit et cboProduit.
Il contient 3 contrôles : Le libellé de la formation, le CodeNiveau nourrit par une liste déroulante enrichie à la volée et la durée de la formation.

III-C-4-a - Propriétés : Données



Pour ce formulaire, nous ne ferons pas de requête puisque tous les champs seront renseignés lors de la saisie. Soit par saisie directe, soit par la liaison champ PERE/ champ FILS.

III-C-4-b - Propriétés : Format

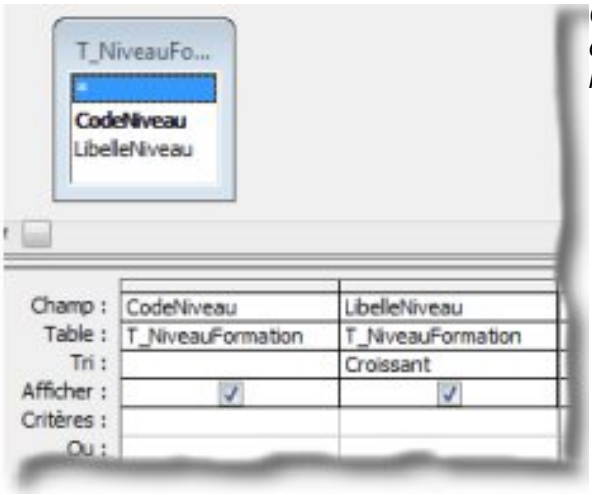


Beaucoup de propriétés modifiées pour épurer le formulaire. Cependant, il est à noter la propriété : Affichage par défaut = Formulaires continus.
Cette propriété agit sur l'affichage de la zone Détail. Ainsi plusieurs enregistrements seront affichés l'un au dessous de l'autre.

III-C-4-c - Les contrôles dans le sous formulaire

III-C-4-c-i - La liste déroulante : CodeNiveau

III-C-4-c-i-1- - La Source de la liste



Comme pour les autres listes, la création de la liste via l'assistant nous génère la requête ci-contre.

On obtient alors le code SQL ci-dessous :

```
SELECT T_NiveauFormation.CodeNiveau, T_NiveauFormation.LibelleNiveau
FROM T_NiveauFormation
ORDER BY T_NiveauFormation.LibelleNiveau;
```

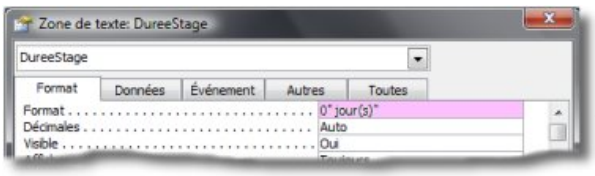
La requête est triée sur le libellé du niveau ce qui fait qu'à chaque ajout de valeur, la liste reste organisée par ordre alphabétique.

III-C-4-c-i-2- - L'évènement "sur Absence dans liste"

```
Private Sub NiveauStage_NotInList(NewData As String, Response As Integer)
            ' Appel de la procédure d'ajout des données
            Call AjoutDansListe("T_NiveauFormation", "LibelleNiveau", NewData)
            ' Mise à jour de la liste
            Response = acDataErrAdded
End Sub
```

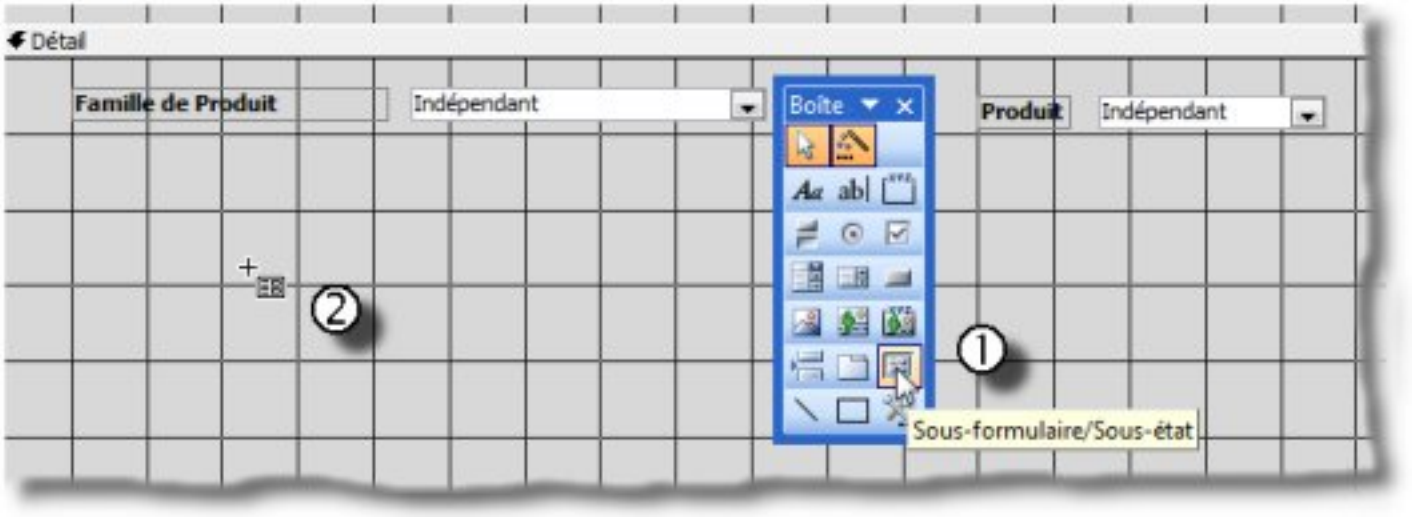
Ce code fait donc appel à la routine d'AjoutDansListe déjà présentée plus haut.

III-C-4-c-ii - Le contrôle DureeStage

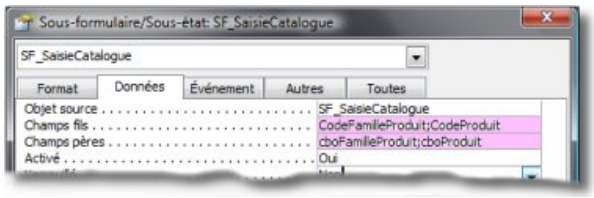


Une seule propriété a été modifiée sur ce contrôle. Il s'agit du format afin d'avoir une présentation plus conviviale lors de la saisie. voir ci-contre.

III-C-4-d - Implantation du sous formulaire "SF_SaisieCatalogue"

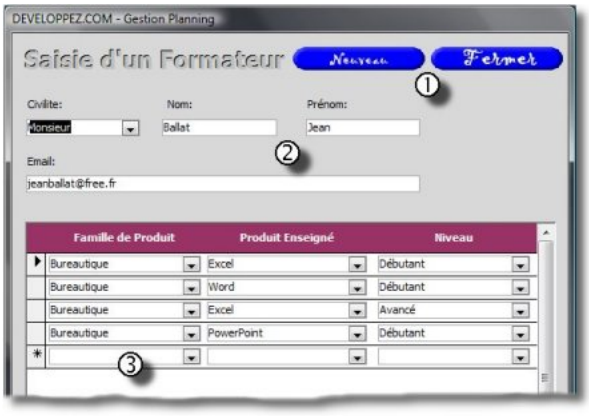


1) Dans la boite à outils, cliquez sur l'outil Sous Formulaire/Sous Etat.
2) Cliquez sur le formulaire à l'endroit où le sous formulaire doit être implanté.
Choisir le sous formulaire à implanter.



Il ne nous reste qu'à créer la liaison entre les listes déroulantes du formulaire principal et le sous formulaire.
Cliquez droit sur le sous formulaire pour afficher les propriétés de celui-ci.
Renseignez les lignes champs PERE et champs Fils.

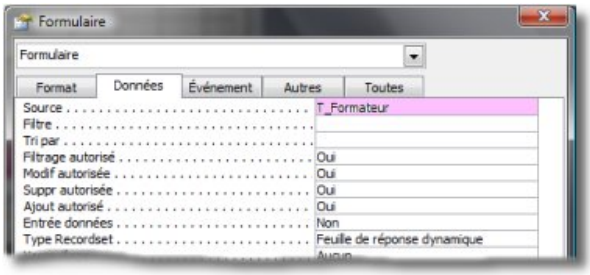
III-D - Le formulaire : F_SaisieFormateur



Objectif du formulaire *:*
Permettre la saisie des formateurs du centre et de leurs compétences.
Fonctionnement *:*
L'utilisateur passe par le bouton "Nouveau" (1) pour saisir un nouvel enregistrement. Il saisit l'identité du formateur et dans le sous formulaire (2), liste l'ensemble des compétences de celui-ci (3).
Les tâches :
Les listes FamilleProduits, Produits, Niveau seront toutes enrichies à la volée. Chaque produit est associé à une couleur que l'on déterminera à la création du produit.
les objets :
Le formulaire principal est attaché à la table T_Formateur.
Le sous formulaire SF_SaisieFormateurCompetences est attaché à la table T_ProduitEnseigne. Il est rattaché au formulaire principal par le CodeFormateur.
Les évènements :
Les trois listes contenues dans le sous formulaire seront mises à jour à la volée. Cela signifie donc que l'on agira sur l'évènement "sur Absence dans liste" de chacune de celles-ci.
Le bouton "btnNouveauS" permettra l'accès à un nouvel enregistrement, et le bouton "btnFermerS" fermera le formulaire. Nous retrouverons donc l'évènement "sur Souris Relâchée" pour les deux boutons.

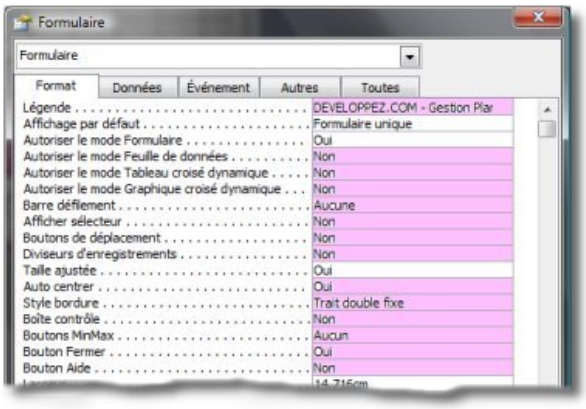
III-D-1 - Les propriétés du formulaire

III-D-1-a - Propriétés : Données



Le formulaire est dépendant de la table T_Formateurs définie dans le chapitre "Conception et Analyse"

III-D-1-b - Propriétés : Format

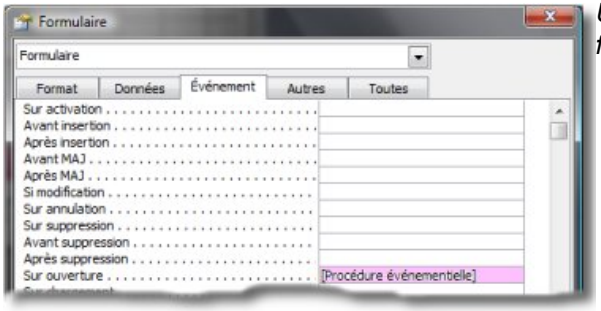


Le formulaire sera paramétré comme les autres formulaires de saisie. Encore une fois tous les boutons système sont supprimés.

III-D-1-c - Propriétés : Autres

Comme pour les autres formulaires de saisie, celui-ci reste au premier plan. Il faut cliquer sur le bouton "FERMER" pour se libérer de la saisie d'un formateur.

III-D-1-d - Propriétés : Evènements

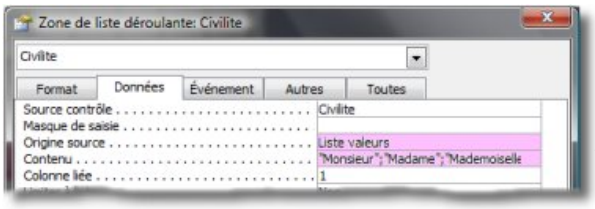


Un seul évènement sera contrôlé sur le formulaire.

L'évènement : **"sur Ouverture"** Il concerne le chargement des images des boutons "Nouveau" et "Fermer".

```
Private Sub Form_Open(Cancel As Integer)
    ' Récupération du formulaire actif
    Set frmActif = Me
    ' Chargement des images
    Me.btnFermerR.Picture = CurrentProject.Path & "\Image\btnFermerR.jpg"
    Me.btnFermerS.Picture = CurrentProject.Path & "\Image\btnFermerS.jpg"
    Me.btnFermerC.Picture = CurrentProject.Path & "\Image\btnFermerC.jpg"
    Me.btnNouveauR.Picture = CurrentProject.Path & "\image\btnNouveauR.jpg"
    Me.btnNouveauS.Picture = CurrentProject.Path & "\image\btnNouveauS.jpg"
    Me.btnNouveauC.Picture = CurrentProject.Path & "\image\btnNouveauC.jpg"
End Sub
```

III-D-2 - Le contrôle Civilité



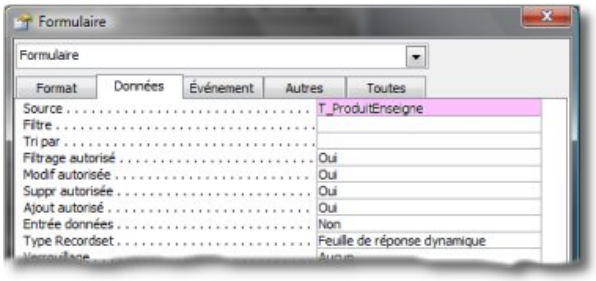
Une particularité pour la source du contrôle dépendant du champ Civilite de la table. Etant donné que la civilité se limite à 3 valeurs, j'ai opté pour la liste de valeurs et non une Table/Requête.

Pour les autres contrôles du formulaire principal, il n'y a aucune particularité.

III-D-3 - Le contrôle Sous Formulaire

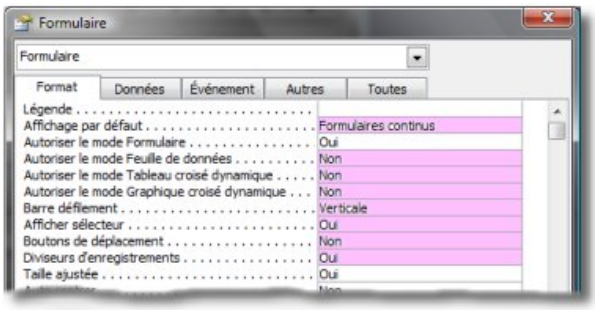
Ce formulaire est relié à **CodeFormateur**, clé primaire de la table T_Formateur.
Il contient 3 listes déroulantes : La famille du produit, le produit, le niveau de la formation.

III-D-3-a - Propriétés : Données



Le formulaire est associé à la table T_Formateurs.

III-D-3-b - Propriétés : Format



On retrouvera les mêmes propriétés modifiées que pour le sous formulaire : "SF_SaisieCatalogue".

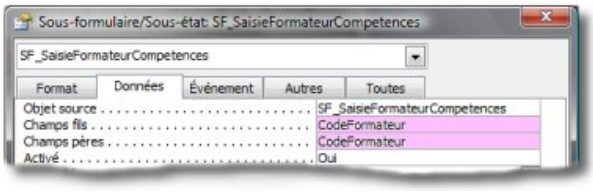
III-D-3-c - Les contrôles dans le sous formulaire

Les 3 listes sont gérées de la même façon que dans le formulaire F_SaisieCatalogue.
Je vous renvoie donc au chapitre correspondant.
Cependant, on notera que la liste des produits enseignés n'est pas dépendante de la liste des familles de produits.
Pourquoi ?
Le sous formulaire est de type "continu".
En conséquence, le fait de changer de famille, une mise à jour automatique de la liste masque les valeurs des autres champs et donne l'impression que les infos sont perdues.

Pour remédier à ce souci, j'ai donc laissé la liste des produits s'afficher dans son intégralité. Charge à l'utilisateur de faire "son marché".
Par contre, il est absolument nécessaire de contrôler le choix.
On agira sur l'évènement "sur sortie" du contrôle. (voir la procédure ci-dessous)

```
Private Sub CodeProduitEnseigne_Exit(Cancel As Integer)
    'Si la liste est nulle sortie de la procédure
    If IsNull(CodeProduitEnseigne.Column(2)) Then Exit Sub
    ' Contrôle de la sélection d'un produit
    If CDec(CodeProduitEnseigne.Column(2)) <> CDec(CodeFamilleProduit) Then
        ' message d'alerte

  MsgBox "Le Produit sélectionné ne correspond pas à la Famille de produits choisie ! " & vbCrLf & _
                  "Veuillez refaire votre choix", vbInformation, cstDVP
        'action sur la liste
        With CodeProduitEnseigne
         ' Efface le contenu de la saisie
            .Value = ""
            ' Ouvre la liste automatiquement
            .Dropdown
        End With
        ' annule l'évènement "Sortie du contrôle"
        Cancel = True
    End If
End Sub
```

III-D-3-d - Implantation du sous formulaire : "SF_ProduitEnseigne"



On remarquera qu'après l'implantation du sous formulaire, il est inutile de créer les liens PERE et FILS, ceux-ci ont été générés automatiquement. Cela a été possible parce que le formulaire principal est attaché à une table et que la clé primaire est en relation avec un champ de la source du sous formulaire.

III-D-4 - Les boutons du formulaire principal

Le principe d'animation étant posé au début de l'article, il nous reste à voir le code correspondant aux boutons "btnNouveauS" et "btnFermerS".
Dans les deux cas, le code est généré sur l'évènement "sur Souris relâchée".

III-D-4-a - Le bouton : Nouveau

```
Private Sub btnNouveauS_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single)
    ' Ajout d'un nouvel enregistrement
    DoCmd.GoToRecord , , acNewRec
    ' Appel de la routine de bascule des boutons
    Call BasculerBouton("btnNouveauC", "btnNouveauS")
End Sub
```

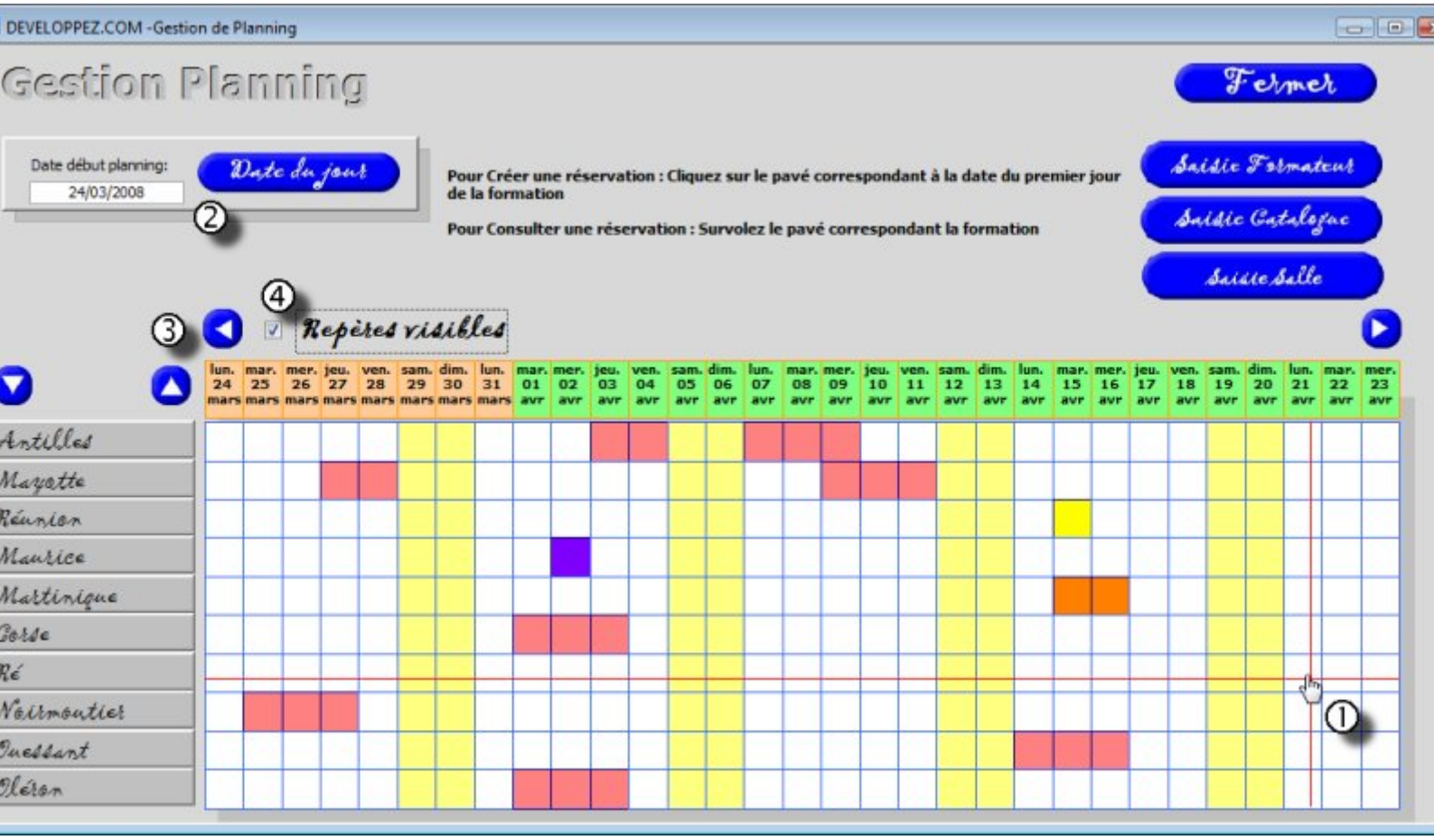
Pas de difficultés particulières, nous sautons sur un nouvel enregistrement et nous rebasculons les boutons.

III-D-4-b - Le bouton : Fermer

```
Private Sub btnFermerS_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single)
    ' Test de fermeture du formulaire
    If MsgBox("Voulez-vous valider les modifications ?", vbQuestion + vbYesNo, cstDVP) = vbYes Then
        DoCmd.Close
    Else
        ' Annule la saisie
        Me.Undo
        ' ferme le formulaire
        DoCmd.Close
    End If
End Sub
```

Il s'agit en fait d'une simple procédure de fermeture de formulaire.

III-E - Le formulaire : F_Planning



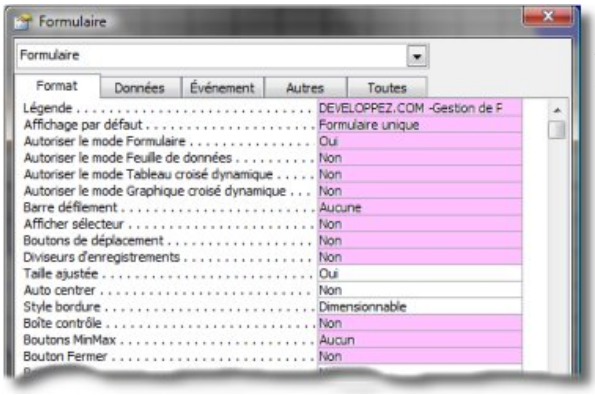
Objectif du formulaire :
Permettre la réservation d'une période de formation.
Fonctionnement :
Ce formulaire représente la plate-forme de l'application. Tous les formulaires seront accessibles à partir de ce niveau.
L'utilisateur survole les plages de planning et clique pour lancer la procédure de réservation. (1)
A l'ouverture, le planning s'ouvrira sur la date du jour. (2) (Remarque : pour avoir automatiquement des données affichées, j'ai figée la date à l'ouverture. Il suffira de modifier la ligne de code associée).
Il permettra de naviguer aussi bien entre les salles (glissement vertical) et dans le temps (glissement horizontal). (3)
Pour faciliter le repérage d'une plage, on utilisera les repères. (4)
Les tâches :
En cliquant sur une plage : ouverture du formulaire de réservation.
En cliquant sur le bouton "Date du jour" : le planning se repositionne automatiquement sur la date du jour.

En cliquant sur les boutons de navigation : le planning "glisse" horizontalement (calendrier perpétuel) ou verticalement (affichage des salles masquées).
En cliquant sur la case à cocher "Repères visibles" : les guides s'affichent ou se masquent.
les objets :
Le formulaire est indépendant et ne contient que des objets dessinés.
Les évènements :
La réservation sera possible à partir de l'évènement "sur Clic" de chaque pavé de la zone "planning".
Le retour à la date du jour se fera sur l'évènement "sur Souris Relâchée" du bouton "btnDateJourS".
La navigation dans le planning se gèrera sur l'évènement "sur Souris Relâchée" des 4 boutons : "btnBasS", "btnHautS", "btnGaucheS", "btnDroiteS".
L'affichage des guides se fera par l'évènement "Après MAJ" de la case à cocher "optReperes".
L'ouverture des différents formulaires s'exécutera sur l'évènement "sur Souris Relâchée" des 3 boutons : "btnFormateurS", "btnCatalogueS", "btnSalleS".

III-E-1 - Les propriétés du formulaire

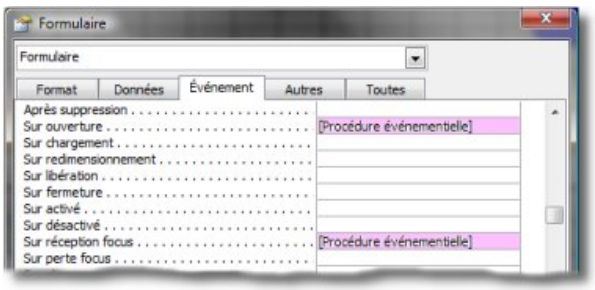
Le formulaire est indépendant, toutes les données sont gérées via le code.

III-E-1-a - Propriétés : Format



Comme pour les autres formulaires, tous les boutons sont désactivés ainsi que les barres de défilement et autre sélecteur.

III-E-1-b - Propriétés : Evènements



Deux évènements seront gérés sur le formulaire.

L'évènement "sur Ouverture".
Tout d'abord, on y retrouve le chargement des images des boutons.

```
Private Sub Form_Open(Cancel As Integer)
    ' Mise en place animation bouton
    Set frmActif = Me
    ' initialisation des images
    btnDroiteR.Picture = CurrentProject.Path & "\image\btnDroiteR.jpg"
    btnDroiteS.Picture = CurrentProject.Path & "\image\btnDroiteS.jpg"
```

```
    btnDroiteC.Picture = CurrentProject.Path & "\image\btnDroiteC.jpg"
    btnGaucheR.Picture = CurrentProject.Path & "\image\btnGaucheR.jpg"
    btnGaucheS.Picture = CurrentProject.Path & "\image\btnGaucheS.jpg"
    btnGaucheC.Picture = CurrentProject.Path & "\image\btnGaucheC.jpg"
    btnDateJourR.Picture = CurrentProject.Path & "\image\btnDateJourR.jpg"
    btnDateJourS.Picture = CurrentProject.Path & "\image\btnDateJourS.jpg"
    btnDateJourC.Picture = CurrentProject.Path & "\image\btnDateJourC.jpg"
    btnFormateurR.Picture = CurrentProject.Path & "\image\btnFormateurR.jpg"
    btnFormateurS.Picture = CurrentProject.Path & "\image\btnFormateurS.jpg"
    btnFormateurC.Picture = CurrentProject.Path & "\image\btnFormateurC.jpg"
    btnFermerR.Picture = CurrentProject.Path & "\image\btnFermerR.jpg"
    btnFermerS.Picture = CurrentProject.Path & "\image\btnFermerS.jpg"
    btnFermerC.Picture = CurrentProject.Path & "\image\btnFermerC.jpg"
    btnCatalogueR.Picture = CurrentProject.Path & "\image\btnCatalogueR.jpg"
    btnCatalogueS.Picture = CurrentProject.Path & "\image\btnCatalogueS.jpg"
    btnCatalogueC.Picture = CurrentProject.Path & "\image\btnCatalogueC.jpg"
    btnSalleR.Picture = CurrentProject.Path & "\image\btnSalleR.jpg"
    btnSalleS.Picture = CurrentProject.Path & "\image\btnSalleS.jpg"
    btnSalleC.Picture = CurrentProject.Path & "\image\btnSalleC.jpg"
    btnBasR.Picture = CurrentProject.Path & "\image\btnBasR.jpg"
    btnBasS.Picture = CurrentProject.Path & "\image\btnBasS.jpg"
    btnBasC.Picture = CurrentProject.Path & "\image\btnBasC.jpg"
    btnHautR.Picture = CurrentProject.Path & "\image\btnHautR.jpg"
    btnHautS.Picture = CurrentProject.Path & "\image\btnHautS.jpg"
    btnHautC.Picture = CurrentProject.Path & "\image\btnHautC.jpg"
```

Ensuite, on procède à l'initialisation des différentes variables déclarées de type "Public" dans un module spécial.

```
' initialisation des variables
    ' Affecte la couleur par défaut de la première étiquette du planning
    ' Ainsi le dessin des étiquettes des dates commence toujours par la même couleur.
    lngCouleurfond = 10079487
    ' désactivation des repères
    optReperes = False
    ' Initialisation de la date
    varDateTraitee = CDate("24/03/08")
    ' affiche la date du jour sur le formulaire
    txtDateDebutPlanning = varDateTraitee
    ' vide la chaine des salles concernées
    strSallesConcernees = ""
    ' Initialisation de la variable fin de jeu d'enregistrements (lecture des entêtes de lignes)
    boolEOF = False

    ' Initialisation du compteur d'enregistrements
    lngCompteurRecord = 0
    lngRecordDepart = 0
    intDefilementSalle = 0

    ' intitialisation des variables dates pour l'extraction des réservations
    varDateDebutPlanning = Format(varDateTraitee, "mm/dd/yy")
    varDateFinPlanning = Format(DateAdd("d", 31, varDateDebutPlanning), "mm/dd/yy")

    ' Affiche le formulaire en plein écran
    DoCmd.Maximize
```

On remarquera l'initialisation de la date du formulaire fixée au 24/03/08.
J'ai opté pour cette solution afin qu'à l'ouverture du planning, les données exemples soient toujours visibles.
Il suffira de remplacer cette ligne par le code suivant pour initialiser le planning sur la date du jour

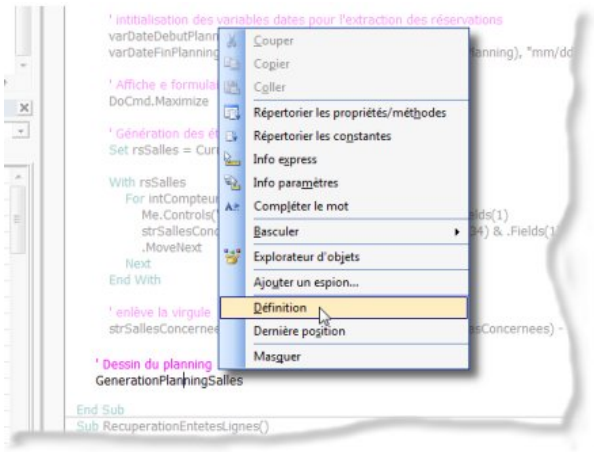
```
    ' Initialisation de la date
    varDateTraitee = Date()
```

Il nous faut donc afficher la totalité des salles du centre qu'il y ait formation prévue ou pas.
Afin de préparer la requête qui va extraire les données toutes salles confondues, je crée une chaîne avec le nom des salles qui va servir de critères avec l'opérateur IN.

```
        ' Génération des étiquettes de salles
        Set rsSalles = CurrentDb.OpenRecordset(cstSalles)

        With rsSalles
            For intCompteur = 1 To 10
                Me.Controls("lblNomSalle" & intCompteur).Caption = .Fields(1)
                strSallesConcernees = strSallesConcernees & "," & Chr(34) & .Fields(1) & Chr(34)
                .MoveNext
            Next
        End With

        ' enlève la virgule en fin de chaîne
        strSallesConcernees = Right(strSallesConcernees, Len(strSallesConcernees) - 1)
```

Un petite remarque sur cette partie de code :
En fin de code, j'enlève la virgule placée en fin de chaîne. En effet, la boucle de traitement ajoute une "," à chaque passage. Si je garde cette virgule, celle-ci va générer une erreur.
Enfin la dernière partie du code fait appel à la routine de création du planning.

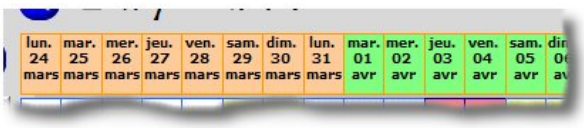
```
    ' Dessin du planning
    GenerationPlanningSalles
```



Pour avoir rapidement accès à un sous programme, il suffit de cliquer droit sur le nom de celui-ci et de choisir "Définition" dans le menu contextuel.
Nous obtenons donc le code suivant :

```
Public Sub GenerationPlanningSalles()
 ' Déclaration de variables locales
    Dim intCompteurColonnes As Integer, intCompteurLignes, strInfoBulle As String, boolReservation As
 Boolean, boolEOFPlanning As Boolean

     ' intitialisation des variables
     boolReservation = False
     boolEOFPlanning = False

    ' Définition des limites du planning glissant (affichage sur 31 jours)
    varDateTraitee = varDateDebutPlanning
    varDateFinPlanning = DateAdd("d", 31, varDateDebutPlanning)
```

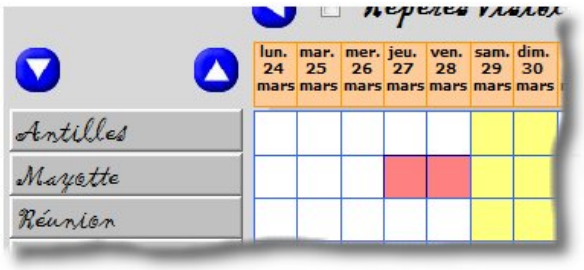
Dans cette première partie du code, je déclare des variables qui seront de portée locale (elles sont déclarées dans la procédure) et on les initialise.
On remarquera également la fonction DateAdd(). Pour plus d'informations sur les fonctions de date, je vous conseille la lecture de : **Les fonctions Date/Heure**



On procède ensuite à la création des étiquettes du planning.

```
    ' Initialisation des étiquettes de date
    For intCompteur = 1 To 31
     ' Génére le contenu de l'étiquette (Propriété : Caption)
        Forms("F_Planning").Controls("lbljour" & intCompteur).Caption =
Format(varDateTraitee, "ddd dd mmm")

        ' Changement couleur de l'étiquette
        If Day(varDateTraitee) = 1 Then
            If lngCouleurfond = 10079487 Then
                lngCouleurfond = 8454016
            Else
                lngCouleurfond = 10079487
            End If
        End If

       ' Colorise l'étiquette de date
        Forms("F_Planning").Controls("lbljour" & intCompteur).BackColor = lngCouleurfond
        ' Incrémentation de la date
        varDateTraitee = DateAdd("d", 1, varDateTraitee)
    Next
```

Durant le process, j'initialise la couleur de la première étiquette afin de pouvoir alterner les couleurs en fonction du changement de mois.
Les étiquettes ont un nom composé de "lblJour" et d'un index correspondant à la position dans la ligne.
Le process continue avec le dessin des zones réservées du planning.



On remarquera que les plages de réservation du planning sont soit blanches, jaunes ou colorées en fonction du produit.
Blanches : les plages sont disponibles.
Jaunes : Les plages sont bloquées : Week-end !
Colorées : Les plages sont réservées. Un survol de la souris et l'info-bulle affiche les données relatives à la formation réservée.

Avant de voir le code, il est nécessaire d'aborder le principe que j'ai appliqué.
Le dessin tourne autour de deux boucles principales :
1) le compteur de lignes : représente les salles.
2) le compteur de colonnes représente les jours du planning de réservation et compare avec les dates du recordset rsPlanning.
Le Recordset "rsPlanning" contient les données issues de la table T_Planning correspondantes à la plage de dates du planning (requête paramétrée).
On remarquera une variable **boolEOFPlanning** qui permettra de terminer le dessin du planning lorsque plus aucune réservation n'a été faite dans la période à afficher.

```
' ************************************************************************
'                           DESSIN DES ZONES RESERVEES
'************************************************************************

   ' Initialisation de la requete
   strSqlWherePlanning = " Where NomSalleFormation IN(" & strSallesConcernees & _
       ") AND DateStage BETWEEN #" & Format(varDateDebutPlanning, "mm/dd/yy") & _
    "# AND #" & Format(varDateFinPlanning, "mm/dd/yy") & "# OR DateStage Is Null" _
                        & " ORDER BY T_SalleFormation.CodeSalleFormation, T_Planning.DateStage"
```

```
        Set rsPlanning = CurrentDb.OpenRecordset(cstPlanning & strSqlWherePlanning)
        ' Ouverture du jeu d'enregistrements
        With rsPlanning
             ' Récupère le nom de la salle réservée dans le premier enregistrement rsPlanning
             strNomSalleEnCours = .Fields(0)
             ' Début de la boucle de traitement des salles
             For intCompteurLignes = 0 To 9
                 ' Début de la boucle de traitement des pavés du planning
                For intCompteurColonnes = 1 To 31
                     ' Incrémentation de la date -> Passe à la colonne suivante de la zone planning
                     varDateTraitee = DateAdd("d", intCompteurColonnes - 1, varDateDebutPlanning)
                     ' Test de contrôle de lecture de la table des données à afficher dans le planning

  ' Si le test est vrai, il n'y a plus de réservation à noter dans la plage de date sélectionnée

  ' On saute donc directement au dessin d'un carré Blanc ou Jaune (si varDateTraitee = date d'un weekend)
                     If boolEOFPlanning = True Then
                         ' Branchement à l'étiquette "Reprise"
                         ' Dessine que les plages blanches ou jaunes
                         GoTo Reprise
                     Else

  ' Test pour la concordance entre la salle en cours de traitement et le nom de la salle concernée par une réserv

  ' Si différent, branchement à "Reprise" pour dessiner les pavés blancs ou jaunes
                         If .Fields(0) <> Forms("F_Planning").Controls("lblNomSalle" &
 intCompteurLignes + 1).Caption Then
                             GoTo Reprise
                         Else

  ' Test de repérage de fin de fichier (lecture des données concernant les réservations
                             If .EOF Then
                                 ' intitialisation de la variable de fin de lecture des réservations
                                 boolEOFPlanning = True
                             End If
                             ' Test de concordance entre la date réservée et la date traitée
                             If .Fields(1) = varDateTraitee Then
                             ' Préparation de l'info bulle
                             strInfoBulle = "Produit : " & .Fields(3) & vbCrLf & "Objet : " &
 .Fields(2) & vbCrLf & _
                                            "Journée en cours : " & .Fields("Quantieme") &
 IIf(.Fields("Quantieme") = 1, "er ", "ème ") _
                                            & vbCrLf & "Formateur : " & .Fields(4)
                                 'Colore le pavé dans la couleur du produit (Réservation salle)
                                 With Forms("F_Planning").Controls("lblindispo" & intCompteurLignes &
 intCompteurColonnes)
                                     .BackColor = rsPlanning.Fields(6)
                                     .ControlTipText = strInfoBulle
                                     .BorderColor = 10485760
                                 End With
                                     ' Passe à l'enregistrement suivant
                                     .MoveNext
                                     strNomSallePrecedent = strNomSalleEnCours
                                     strNomSalleEnCours = .Fields(0)

  ' Intitialisation de la variable qui indique qu'une réservation est en cours de traitement
                                     ' Permet de passer outre le traitement de la zone "Reprise"
                                     boolReservation = True
                             End If
Reprise:
                             If boolReservation = False Then
                             ' Recherche si la date en cours de traitement est un jour de weekend
                             If Weekday(varDateTraitee) = vbSaturday Or Weekday(varDateTraitee) =
 vbSunday Then
                                 With Forms("F_Planning").Controls("lblindispo" & intCompteurLignes &
 intCompteurColonnes)
                                     ' affecte la couleur réservée au weekend
                                     .BackColor = 8454143
                                     ' Efface le contenu de l'info bulle du pavé du planning
                                     .ControlTipText = ""
                                     ' Réaffecte la couleur de bordure du pavé
                                     .BorderColor = 16737843
```

```
                                End With
                            Else
                                ' Ce n'est pas un jour de Weekend. Remise à blanc du pavé
                                With Forms("F_Planning").Controls("lblindispo" & intCompteurLignes &
 intCompteurColonnes)
                                    .BackColor = 16777215
                                    .ControlTipText = ""
                                    .BorderColor = 16737843
                                End With
                            ' Fin du test sur date traitée
                            End If
                            ' Fin test sur journée réservée
                            End If
                        End If
                    End If
                    ' Réinitialisation de la variable de réservation
                    boolReservation = False
                Next
            Next
        End With
End Sub
```

L'évènement : **"Sur Réception focus"**
Une petite ligne de code pour réinitialiser la variable **frmActif**

```
Private Sub Form_GotFocus()
 ' récupération du formulaire en cours d'utilisation (Gestion de l'animation des boutons)
    Set frmActif = Me
End Sub
```

III-E-1-c - La section Détail du formulaire

Il est bien sûr nécessaire, pour gérer l'animation des boutons et du pointeur de souris, d'écrire la procédure ci dessous.

```
Private Sub Détail_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)
    ' Récupération du formulaire actif
    Set frmActif = Me

    ' réintialisation des images
    btnDroiteR.Visible = True
    btnDroiteS.Visible = False
    btnDroiteC.Visible = False
    btnGaucheR.Visible = True
    btnGaucheS.Visible = False
    btnGaucheC.Visible = False
    btnDateJourR.Visible = True
    btnDateJourS.Visible = False
    btnDateJourC.Visible = False
    btnFormateurR.Visible = True
    btnFormateurS.Visible = False
    btnFormateurC.Visible = False
    btnCatalogueR.Visible = True
    btnCatalogueS.Visible = False
    btnCatalogueC.Visible = False
    btnSalleR.Visible = True
    btnSalleS.Visible = False
    btnSalleC.Visible = False
    btnBasR.Visible = True
    btnBasS.Visible = False
    btnBasC.Visible = False
    btnHautR.Visible = True
    btnHautS.Visible = False
    btnHautC.Visible = False

    ' réintialisation de la souris
    Screen.MousePointer = 0
```

```
End Sub
```

III-E-2 - Le contrôle : Plage du planning

En fait la plage du planning est composée de 310 plages sensibles (10 lignes sur 31 colonnes). Chacune porte un nom composé du radical : "lblIndispo", d'un numéro de ligne commençant par 0 et d'un n° de colonne commençant par 1. Ce procédé permettra de reconnaitre quelle salle sera sélectionnée et à quelle date.
Ces zones seront régies par deux évènements: 1) "Sur Souris déplacée" : cela permettra de faire suivre les guides pour faciliter la lecture sur la plage.
2) "Sur clic" : cela permettra de réserver une date pour une formation.
L'évènement : **"sur Souris déplacée"**.
On trouvera le code qui gère le suivi des repères.

```
Private Sub lblIndispo01_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)
 ' Récupération du contrôle en cours
    ctrlLabel = "lblIndispo01"
    ' Gestion du Changement de pointeur de souris
    Call ChangeMouseToHand
    ' Gestion de suivi des guides.
    Call TracerRepere(ctrlLabel, X, Y)
End Sub
```

Cette séquence d'instructions se retrouvera sur l'ensemble des contrôles composant la plage du planning.
Voyons en détail chacun des appels aux sous programmes.

```
 '************************************
    ' Code Proposé par Faw - Modérateur DVP
    '************************************

    ' Déclaration des variables
    Private Const IDC_HAND = 32649
    Private Declare Function LoadCursor Lib "user32" Alias "LoadCursorA" (ByVal hInstance As
 Long, ByVal lpCursorName As Long) As Long
    Private Declare Function SetCursor Lib "user32" (ByVal hCursor As Long) As Long

Function ChangeMouseToHand()
    Dim hCur As Long
    hCur = LoadCursor(0, IDC_HAND)
    If (hCur > 0) Then
        SetCursor hCur
    End If
End Function
```

Cette procédure sera créée dans un module séparé (voir mod_APISouris).
Objectif : Changer le pointeur de la souris (Retrouver cette procédure dans la **http://access.developpez.com/faq/?page=Ctrl#survol_ctrl**)

```
Sub TracerRepere(ByVal ctrlLabel As String, PositionX, PositionY)
 ' Controle si l'option "Afficher les repères" est active ou non
 If optReperes = True Then
     ' Génére la position du trait en fonction du contrôle survolé
     objRepereH.Top = Me.Controls(ctrlLabel).Top + PositionY + 50
     objRepereV.Left = Me.Controls(ctrlLabel).Left + PositionX + 50
     ' Redessine le formulaire
     Me.Repaint
 End If
End Sub
```

On remarquera la constante "+ 50" qui a pour effet de décaler le trait par rapport à la pointe de la souris, sinon le clic n'est pas sur la plage mais sur le trait lui-même.

L'évènement : **"sur Clic"**
Il permettra de créer une réservation après contrôle de la zone.
Ce contrôle se fait à plusieurs niveaux :
- La date choisie n'est pas un jour de weekend.
- La durée de la formation n'est pas coupée par un weekend.
- Une date est déjà réservée dans la plage de la formation.

```
Private Sub lblIndispo01_Click()
    strNomPave = "lblIndispo01"
    ' Appel de la routine de contrôle de réservation
    Reservation
End Sub
```

Voyons plus en détail la routine "Réservation".

```
Sub Reservation()
    ' Récupère le nom et le n° de code de la salle cliquée
    RecuperationSalle
    ' Vérifie si la date sélectionnée n'est pas un WeekEnd et si la salle n'est pas déjà occupée
    ControleDispoSalle
    ' Redessine le Planning
    If boolReservationPossible = True Then
        GenerationPlanningSalles
    End If
End Sub
```

On constate donc que la procédure fait appel à trois routines.
1) Récupérer, grâce au clic, le nom de la salle cliquée.

```
Sub RecuperationSalle()
    ' Recuperation du nom de la salle cliquee
    intNumLigne = Mid(strNomPave, 11, 1) + 1
    ' recupération du nom de la salle
    strSalle = Me.Controls("lblNomSalle" & intNumLigne).Caption
    ' Recupération du N° de la Salle
    Set rsSalles = CurrentDb.OpenRecordset(cstSalles & " WHERE NomSalleFormation = '" & strSalle & "'")
    lngCodeSalleFormation = rsSalles.Fields(0)
    rsSalles.Close
End Sub
```

2) Contrôler la validité de la date cliquée.

```
Sub ControleDispoSalle()
    ' Récupération de la date cliquée
    varDateCliquee = DateAdd("d", Mid(strNomPave, 12, Len(strNomPave) - 11) - 1, varDateDebutPlanning)
    ' test sur la couleur du weekend
    If Forms("F_Planning").Controls(strNomPave).BackColor = 8454143 Then
        MsgBox "On ne travaille pas le Week End !", vbInformation, cstDVP
    ' test sur la couleur de réservation
    ElseIf Forms("F_Planning").Controls(strNomPave).BackColor <> 16777215 Then
        MsgBox "La Salle est déjà réservée !", vbInformation, cstDVP
    Else
        ' Ouverture du formulaire de Réservation d'une formation
        DoCmd.OpenForm "F_Reservation"
    End If
End Sub
```

3) Redessiner le planning avec la nouvelle réservation.

III-E-3 - Le contrôle : Repères visibles

Petite option de confort pour, en fonction des besoins, afficher ou non les guides.
Un seul évènement est utilisé : "Après MAJ"

```
Private Sub optReperes_AfterUpdate()
    If optReperes = True Then
        'affiche les reperes du planning
        objRepereH.Visible = True
        objRepereV.Visible = True
    Else
        ' Masque les reperes du planning
        objRepereH.Visible = False
        objRepereV.Visible = False
    End If
End Sub
```

III-E-4 - Le bouton : Date du Jour

Ce bouton aura pour simple fonction de ramener le planning à la date du jour.
Un seul évènement est utilisé : "sur Souris déplacée" (Ce n'est pas un bouton traditionnel mais une image).

```
Private Sub btnDateJourS_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single)
 'Retour à la date du jour
 txtDateDebutPlanning = Date
 varDateDebutPlanning = Date
 ' Dessine le planning
 GenerationPlanningSalles
 ' appel de la routine de bascule des boutons
 Call BasculerBouton("btnDateJourC", "btnDateJourS")
End Sub
```

III-E-5 - Le contrôle : Date Planning

Cette zone de texte indépendante permet de recevoir une date quelconque saisie par l'utilisateur.
Un seul évènement est utilisé : "Après MAJ"

```
Private Sub txtDateDebutPlanning_AfterUpdate()
    ' conversion au type date de la valeur saisie par l'utilisateur
    varDateDebutPlanning = CDate(txtDateDebutPlanning)
    ' Relance le dessin du planning en fonction de la nouvelle date saisie
    GenerationPlanningSalles
End Sub
```

III-E-6 - Le contrôle : Bouton de navigation vers la droite

Il s'agit du bouton : btnDroiteS.
Bouton qui permet d'avancer dans le calendrier perpétuel en redessinant le planning en fonction des dates posées.

```
Private Sub btnDroiteS_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single)
    ' incrémentation de la date traitée de 1 jour
    varDateDebutPlanning = DateAdd("d", 1, varDateDebutPlanning)
    ' Dessine le planning
    GenerationPlanningSalles
    ' appel la routine de bascule des boutons
    Call BasculerBouton("btnDroiteC", "btnDroiteS")
End Sub
```

III-E-7 - Le contrôle : Bouton de navigation vers la gauche

Il s'agit du bouton : btnGaucheS.
A l'instar du contrôle précédent, permet de "revenir dans le passé".

```
Private Sub btnGaucheS_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single)
    ' décrémentation de la date traitée
    varDateDebutPlanning = DateAdd("d", -1, varDateDebutPlanning)
    ' Dessine le planning
    GenerationPlanningSalles
    ' appel la routine de bascule des boutons
    Call BasculerBouton("btnGaucheC", "btnGaucheS")
End Sub
```

III-E-8 - Le contrôle : Bouton de navigation vers le bas

Il s'agit du bouton : btnBasS.
Il permettra d'afficher les 10 valeurs suivantes de la table T_Salles.
Tout comme les autres boutons de navigation, l'évènement géré sera : "sur Souris Déplacée".

```
Private Sub btnBasS_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single)
    ' appel la routine de bascule des boutons
    Call BasculerBouton("btnBasC", "btnBasS")

    ' incrémentation de la boucle de défilement de la liste des salles
    intDefilementSalle = intDefilementSalle + 1

    ' Contrôle de fin de fichier
    If boolEOF = True Then
        MsgBox "Vous êtes en Fin de Fichier", vbInformation, "DEVELOPPEZ.COM - Gestion de Planning"
        intDefilementSalle = intDefilementSalle - 1
        Exit Sub
    End If

    ' Appel de la routine qui permet le repositionnement dans le jeu d'enregistrements
    RepositionnementSalles
    ' Redessine le planning en fonction des nouvelles salles affichées.
    GenerationPlanningSalles
End Sub
```

Le compteur intDefilementSalle permet de compter le nombre de fois que l'utilisateur à cliquer sur le bouton.
Ce compteur permettra de se repositionner dans le jeu d'enregistrements.

```
Sub RepositionnementSalles()
    ' Remise à blanc des Noms de Salles
    For intCompteurPave = 1 To 10
        Me.Controls("lblNomSalle" & intCompteurPave).Caption = " "
    Next

    ' Repositionnement dans le jeu des salles
    lngRecordDepart = intDefilementSalle * 10

    ' Récupération des nouveaux noms à afficher
    RecuperationEntetesLignes

End Sub
```

Cette routine appelle à son tour le sous programme qui permettra de recréer les entêtes

```
Sub RecuperationEntetesLignes()
        ' Réinitialisation des variables
        intCompteur = 1
        strSallesConcernees = ""
        ' Réinstanciation du jeu d'enregistrements
        Set rsSalles = CurrentDb.OpenRecordset(cstSalles)

  ' Relecture du jeu rsSalles pour créer la chaîne qui permettra l'extraction des données correspondantes dans rs
            ' Retour au début du jeu d'enregistrements
            .MoveFirst
            ' Saute au premier enregistrement concerné
            .Move lngRecordDepart
            ' Boucle jusqu'à la fin du jeu d'enregistrements
            Do While Not .EOF
                Me.Controls("lblNomSalle" & intCompteur).Caption = .Fields(1)
                ' Génération de la chaîne de critères qui sera utilisée avec l'opérateur IN
                strSallesConcernees = strSallesConcernees & "," & Chr(34) & .Fields(1) & Chr(34)
                intCompteur = intCompteur + 1
                ' Valeur butoir pour sortir de la boucle
                If intCompteur > 10 Then
                    Exit Do
                Else
                    .MoveNext
                End If
            Loop
            ' Inititlaisation de la variable de fin de fichier
            If .EOF Then
                boolEOF = True
            End If
        End With

        ' enlève la virgule
        strSallesConcernees = Right(strSallesConcernees, Len(strSallesConcernees) - 1)
End Sub
```

III-E-9 - Le contrôle : Bouton de navigation vers le haut

Il s'agit du bouton : btnHautS.
Ce bouton permettra de remonter dans la liste des salles.
L'évènement concerné sera donc : "sur Souris déplacée"

```
Private Sub btnHautS_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single)
    On Error Resume Next
    ' appel la routine de bascule des boutons
    Call BasculerBouton("btnHautC", "btnHautS")

    ' décrémentation de la boucle de défilement de la liste des salles
    intDefilementSalle = intDefilementSalle - 1

    ' libération de la fin du jeu d'enregistrements
    boolEOF = False

    ' Controle de début de jeu d'enregistrements par l'intermédiaire du compteur de défilement
    If intDefilementSalle < 0 Then
       MsgBox "Vous êtes en début de fichier", vbInformation, "DEVELOPPEZ.COM - Gestion de Planning"
        intDefilementSalle = 0
    End If

    lngRecordDepart = intDefilementSalle * 10
    '  Repositionnement dans le jeu d'enregistrements
    RecuperationEntetesLignes

    ' repositionnement en début du jeu des réservations
    rsPlanning.MoveFirst
    lngRecordDepart = 0

    ' Relance le dessin du planning
    GenerationPlanningSalles
```

```
End Sub
```

On remarquera la décrémentation du compteur intDefilementSalle qui permettra de se repositionnement dans le jeu rsSalles

III-E-10 - Les contrôles des autres boutons

Les autres boutons nous permettent donc d'ouvrir, à partir du planning, les différents formulaires de saisie.

1) Saisie Salle. Bouton concerné : btnSaisieSalleS.

```
Private Sub btnSalleS_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single)
    ' ouvre le formulaire de saisie des salles
    DoCmd.OpenForm "F_SaisieSalles"

    ' appel la routine de Bascule des boutons
    Call BasculerBouton("btnSalleC", "btnSalleS")
End Sub
```

2) Saisie formateur. Bouton concerné : btnSaisieFormateurS.

```
Private Sub btnFormateurS_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single)
    ' appel la routine de bascule des boutons
    Call BasculerBouton("btnFormateurC", "btnFormateurS")
    ' Appel du formulaire de saisie d'un nouveau formateur.
    DoCmd.OpenForm "F_SaisieFormateur"
End Sub
```

3) Saisie Catalogue. Bouton concerné : btnSaisieCatalogueS.

```
Private Sub btnCatalogueS_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single)
    ' appel de la routine de bascule des boutons
    Call BasculerBouton("btnCatalogueC", "btnCataloguesS")
    ' Appel du formulaire de saisie d'un nouvel item du catalogue
    DoCmd.OpenForm "F_SaisieCatalogue"
End Sub
```

4) Fermer. Bouton concerné : btnFermerS.

```
Private Sub btnFermerS_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single)
    ' Ferme le formulaire de gestion Planning
    DoCmd.Close
End Sub
```

On pourrait à ce niveau là, fermer le formulaire et quitter l'application. Il faudrait alors modifier le code comme suit:

```
Private Sub btnFermerS_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single)
    ' Ferme l'application de gestion Planning
    DoCmd.Quit
End Sub
```

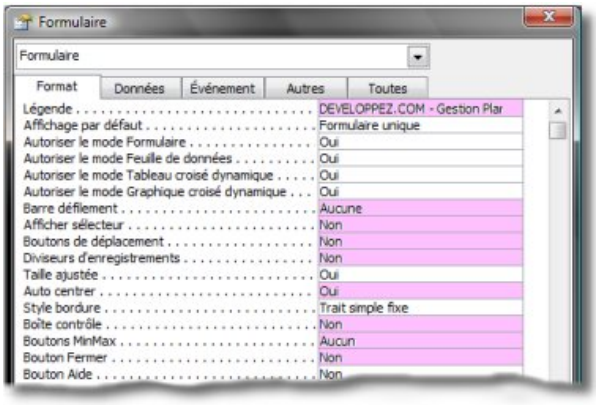
III-F - Le formulaire : F_Reservation

Objectif du formulaire :
Permettre l'affectation d'un produit, d'un formateur et d'un sujet de formation.
Fonctionnement :
Ce formulaire fonctionne sur une imbrication de listes.
Après avoir choisi une famille de produit (1), la liste des produits associés est rafraîchie.
Après avoir sélectionné un produit (2), l'ardoise se met à jour et affiche tous les formateurs susceptibles de faire la formation (3).
La combinaison de la famille de produits et du produit nous permet d'afficher les sujets de stage concernés (4).
La sélection d'un sujet réactualisera l'ardoise pour n'afficher que les formateurs disponibles dans la période de stage.
Le bouton "Valider" provoquera l'enregistrement des données et la génération du planning.
Les tâches :
En mettant à jour la liste cboFamilleProduit : Rafraichissement de la liste cboProduits.
En choisissant un produit : Rafraîchissement de la liste des formateurs compétents et affichage des sujets de formations.
En cliquant sur un sujet de formation : la liste des formateurs compétents se réactualise pour n'afficher que la liste des formateurs disponibles.
En cliquant sur le bouton "Valider" : les tables T_Stages et T_Planning sont renseignées et le planning est redessiné en conséquence.
les objets :
Le formulaire est indépendant et ne contient que des objets dessinés.
Les évènements :
Le rafraichissement de la liste "cboProduits" se fera qu'"Après MAJ" de la liste "cboFamilleProduit".
L'ouverture de la liste "cboProduits" se fera sur "Réception focus" de la liste concernée.
Le rafraichissement de la liste des formateurs compétents se fera "Après MAJ" de la liste "cboProduits".
Le rafraichissement de la liste des formateurs disponibles se fera "sur Clic" du contrôle LibelleStage du sous formulaire.
La validation de la réservation sur fera "sur Souris déplacée" du bouton "btnValiderS".
La fermeture du formulaire se fera "sur Souris déplacée" du bouton "btnFermerS".

III-F-1 - Les propriétés du formulaire

Le formulaire principal est indépendant et le sous formulaire est géré par ses propriétés : champs PERE et FILS

III-F-1-a - Propriétés : Format



Comme pour les autres formulaires de saisie, les boutons ont été désactivés et le formulaire centré.

III-F-1-b - Propriétés : Autres

A l'instar des autres formulaires, celui-ci restera au premier plan jusqu'à clic sur le bouton "btnFermerS"

III-F-1-c - Propriétés : Evènements

Un seul évènement pour ce formulaire sera concerné.
L'évènement **"sur Ouverture"**
Il gèrera le chargement des images et la récupération des valeurs à afficher.

```
Private Sub Form_Open(Cancel As Integer)
    'Récupération des images
    Set frmActif = Me
    Me.btnFermerR.Picture = CurrentProject.Path & "\Image\btnFermerR.jpg"
    Me.btnFermerS.Picture = CurrentProject.Path & "\Image\btnFermerS.jpg"
    Me.btnFermerC.Picture = CurrentProject.Path & "\Image\btnFermerC.jpg"
    Me.objArdoise.Picture = CurrentProject.Path & "\Image\Ardoise.jpg"
    Me.btnValiderR.Picture = CurrentProject.Path & "\Image\btnValiderR.jpg"
    Me.btnValiderS.Picture = CurrentProject.Path & "\Image\btnValiderS.jpg"
    Me.btnValiderC.Picture = CurrentProject.Path & "\Image\btnValiderC.jpg"

    ' Initialise les champs indépendants avec les valeurs sélectionnées
    Me.txtSalleFormation = strSalle
    Me.txtDebutFormation = varDateCliquee
End Sub
```

III-F-2 - Les contrôles du formulaire

III-F-2-a - La liste : cboFamilleProduits

Cette liste alimentée par la table "T_FamilleProduit" permettra l'actualisation de la liste "cboProduits" en fonction de la famille choisie.
L'évènement concerné sera donc "Après MAJ".

```
Private Sub cboFamilleProduit_AfterUpdate()
    ' Mise à jour de la liste des produits disponibles dans la famille sélectionnée
    Me.cboProduits.Requery
    ' Renvoie le focus dans le contrôle "cboProduits"
    DoCmd.GoToControl "cboProduits"
End Sub
```

III-F-2-b - La liste : cboProduits

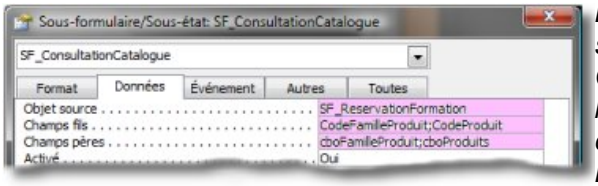
Cette dépendante de la précédente se verra gérer deux évènements :
1) "sur Réception focus" : pour ouvrir la liste déroulante.
2) "Après MAJ" : pour mettre à jour la liste des formateurs compétents.

```
Private Sub cboProduits_AfterUpdate()
    ' Après sélection d'un produit, affiche tous les formateurs susceptibles d'assurer la formation.
    With Me.lstFormateur
        ' Précise le type de source de la liste
        .RowSourceType = "Table/Requête"
        ' Indique le nom de la source de la liste
        .RowSource = cstFormateursCompetents
        ' Rafraichit la liste en fonction de la nouvelle source
        .Requery
    End With
End Sub
```

III-F-2-c - La liste : lstFormateur

Cette liste est gérée par code et ne supporte pas d'évènement. Elle est placée en premier plan au-dessus de l'ardoise.

III-F-2-d - Le sous formulaire : SF_ConsultationCatalogue



La mise à jour du contenu du formulaire sera liée aux champs PERE et FILS. Cependant un évènement est déclenché lorsqu'un choix est fait en cliquant sur le contrôle "Stage" affichant le sujet choisi. L'objet de cet évènement : contrôler la validité de la date sélectionnée et mettre à jour la liste "lstFormateur" en fonction des formateurs disponibles pour la date choisie.

```
Private Sub Stage_Click()
 ' Déclaration des variables
    Dim rsIndisponibles As DAO.Recordset, rsDisponibles As DAO.Recordset
    Dim strDureeStage As String, strSqlWhere As String, strFormateursIndispo As String,
 strFormateursDispo As String
    Dim varDateDebutFormation, varDateFinFormation

    ' Initialisation de la variable de réservation
    boolReservationPossible = True

    ' Mise en forme de la chaine strDureeStage pour respecter le pluriel ou le singulier
    If Me.DureeStage > 1 Then
        strDureeStage = Me.DureeStage & " jours"
    Else
        strDureeStage = Me.DureeStage & " jour"
    End If

    ' Récupération de l'intitulé du stage
```

```
    Forms!F_Reservation!txtFormationSelectionnee = Me.Stage & " - Durée : " & strDureeStage


  ' Récupération des bornes (dates entre lesquelles un ou plusieurs formaturs pourraient être occupés)
    varDateDebutFormation = varDateCliquee
    varDateFinFormation = DateAdd("d", DureeStage - 1, varDateDebutFormation)

    ' Mise à jour de la liste des formateurs disponibles pour le produit sélectionné
    ' Critères permettant de récupérer les formateurs occupés pendant la période de formation
    strSqlWhere = " WHERE  T_ProduitEnseigne.CodeProduitEnseigne = " & lngCodeProduit _
                          & " AND DateStage BETWEEN #" & Format(varDateDebutFormation, "mm/dd/yy")_
        & "# AND #" & Format(varDateFinFormation, "mm/dd/yy") & "#"


    ' Inititalisation du jeu d'enregistrements
    Set rsIndisponibles = CurrentDb.OpenRecordset(cstFormateursDispo & strSqlWhere)

    ' lecture du jeu résultat et création de la chaîne correspondante aux formateurs
 ' occupés afin de les exclure par l'opérateur NOT IN
    With rsIndisponibles
        Do While Not .EOF
            strFormateursIndispo = strFormateursIndispo & "," & .Fields(0)
            .MoveNext
        Loop
    End With

    ' Suppression de la virgule superflue en début de chaîne
    If strFormateursIndispo <> "" Then
        strFormateursIndispo = Right(strFormateursIndispo, Len(strFormateursIndispo) - 1)
        ' Reconstruction de la clause Where d'exclusion des formateurs indisponibles
        strFormateursDispo =
 cstFormateursCompetents & " WHERE T_ProduitEnseigne.CodeProduitEnseigne = " & _
        lngCodeProduit & " AND T_Formateur.CodeFormateur NOT IN (" _
        & strFormateursIndispo & ")"

' ************************************* Alimentation de lstFormateur par une liste de valeurs ***************
        ' Création de la liste de valeurs à afficher dans la liste lstFormateur
        Set rsDisponibles = CurrentDb.OpenRecordset(strFormateursDispo)

        ' Vérification qu'il existe bien des enregistrements dans le jeu résultat
        If rsDisponibles.RecordCount > 0 Then
            ' Réinitialisation de la variablepour réutilisation ci-dessous
            strFormateursDispo = ""


  ' boucle permettant de générer la liste de valeurs à afficher dans le contrôle "lstFormateur"
            With rsDisponibles
                Do While Not .EOF
                    strFormateursDispo = strFormateursDispo & ";" & .Fields(0) & ";" & .Fields(1)
                    .MoveNext
                Loop
             End With

            ' suppression de la virgule superflue en début de chaine
            strFormateursDispo = Right(strFormateursDispo, Len(strFormateursDispo) - 1)
        End If
' ********************************************************************************************

    Else

' ************************** Alimentation de lstFormateur par une requête *****************
        ' initialisation de la variable avec affectation du nom d'une requête existante.
         strFormateursDispo = "R_FormateursProduitsEnseignes"
    End If

    ' Récupération des données à afficher
    With Forms!F_Reservation!lstFormateur
        If strFormateursIndispo = "" Then
            .ColumnCount = 4
            .ColumnWidths = "0;2800;0;0"
            .RowSourceType = "Table/Query"
            .RowSource = strFormateursDispo
```

```
        Else
            .ColumnCount = 2
            .ColumnWidths = "0;2800"
            .RowSourceType = "Value List"
            .RowSource = strFormateursDispo
        End If
    End With

    ' Appel de la fonction de validation de la réservation
    If ControleReservation(varDateCliquee, Me.DureeStage) = True Then
        lngCodeCatalogue = Me.CodeCatalogue
        intDureeStage = Me.DureeStage
        boolFermer = False
        Forms!F_Reservation!btnValiderS.Visible = True
    Else
        DoCmd.Close acForm, "F_Reservation"
    End If
End Sub
```

J'ai voulu, dans cette procédure, montrer les différentes façons d'alimenter une zone de liste.
*On pourra retrouver des informations complémentaires sur les manipulations de liste dans la **F.A.Q.***

III-F-2-e - Le bouton : btnFermerS

Comme pour tous les boutons "image" de l'application, nous passerons par la gestion de l'évènement "sur Souris déplacée".
Objet : fermer le formulaire.

```
Private Sub btnFermerS_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single)
    DoCmd.Close
End Sub
```

III-F-2-f - Le bouton : btnValiderS

Bouton image, comme précédemment, nous affecterons une procédure à l'évènement "sur Souris déplacée".
Objet :
1) Contrôler que la saisie est complète

```
Private Sub btnValiderS_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single)
    Dim strSqlAjoutReservation As String
    Dim varDateStage As Date

    ' Contrôle avant de lancer la réservation
    ' Vérifie si un formateur a été sélectionné
    If IsNull(lstFormateur) Then
        MsgBox "Choisir un formateur avant de valider !", vbInformation, cstDVP
        Exit Sub
    End If

    ' Contrôle d'existence d'un choix de titre de formation
    If lngCodeCatalogue = 0 Then
        MsgBox "Choisir une formation dans le catalogue avant de valider !", vbInformation, cstDVP
        Exit Sub
    End If
```

2) Renseigner les deux tables T_Stages et T_Planning.

```
    ' Désactive les messages système
```

```
    DoCmd.SetWarnings False

    ' Ajout de la formation dans la table des stages

 strSqlAjoutReservation = "INSERT INTO T_Stages ( DateStageDebut, CodeSalleFormation, CodeCatalogue, CodeFormateu
 _
    & "SELECT #" & Format(varDateCliquee, "mm/dd/yy") & "#," & lngCodeSalleFormation _
 & "," & lngCodeCatalogue & "," & lstFormateur

    ' Exécute la requête SQL d'ajout
    DoCmd.RunSQL (strSqlAjoutReservation)
    lngCodeStage = DMax("CodeStage", "T_Stages")

    ' Boucle de création des jours réservés
    For intCompteur = 0 To intDureeStage - 1
        varDateStage = DateAdd("d", intCompteur, varDateCliquee)
        strSqlAjoutReservation = "INSERT INTO T_Planning(CodeStage, DateStage, Quantieme) " & "SELECT "
 _
  & lngCodeStage & ", #" & Format(varDateStage, "mm/dd/yy") & "# ," & intCompteur + 1

        ' Exécute la requête SQL d'ajout
        DoCmd.RunSQL (strSqlAjoutReservation)
    Next

    ' Réactivation des messges systéme
    DoCmd.SetWarnings True
    ' fermeture du formulaire
    DoCmd.Close
```

3) Redessiner le planning.

```
    'Réactivation du dessin du planning
    GenerationPlanningSalles
End Sub
```

IV - LES DECLARATIONS DE VARIABLES

Afin d'alléger le code, j'ai déclaré des constantes reprenant les requêtes qui seront utilisées dans l'application. C'est un choix personnel.
Voici donc ci-dessous les variables utilisées :

```
' Récupération du formulaire en cours d'utilisation pour animation des boutons
Public frmActif As Form

' Déclarations des variables jeux d'enregistrements
Public rsSalles As DAO.Recordset, rsSallesReservees As DAO.Recordset
Public rsPlanning As DAO.Recordset

' Déclarations des constantes
'-----------------------------------
' récupération des données du planning

Public Const
 cstPlanning As String = "SELECT T_SalleFormation.NomSalleFormation, T_Planning.DateStage, " _
                  & "IIf(IsNull([IntituleStage]),Null,
[IntituleStage] &  "" Niveau : "" & [LibelleNiveau]) AS Stage, " _

  & "T_Produits.NomProduit, [Civilite] & "" "" & [NomFormateur] & "" "" & [PrenomFormateur] AS Formateur, "
 _
                  & "T_Catalogue.DureeStage, T_Produits.CodeCouleur, T_Planning.Quantieme " _
                  & "FROM T_SalleFormation INNER JOIN ((T_Produits " _
                  & "INNER JOIN (T_NiveauFormation " _
                  & "INNER JOIN (T_Formateur " _
                  & "INNER JOIN (T_Catalogue " _
                  & "INNER JOIN T_Stages " _
                  & "ON T_Catalogue.CodeCatalogue = T_Stages.CodeCatalogue) " _
```

```
                    & "ON T_Formateur.CodeFormateur = T_Stages.CodeFormateur) " _
                    & "ON T_NiveauFormation.CodeNiveau = T_Catalogue.CodeNiveau) " _
                    & "ON T_Produits.CodeProduit = T_Catalogue.CodeProduit) " _
                    & "INNER JOIN T_Planning ON T_Stages.CodeStage = T_Planning.CodeStage)" _
                    & "ON T_SalleFormation.CodeSalleFormation = T_Stages.CodeSalleFormation"

Public Const cstFormateursCompetents   As String = "SELECT DISTINCT T_Formateur.CodeFormateur, " _
                    & "[Civilite] & "" "" & [NomFormateur] & "" "" & [PrenomFormateur] AS Formateur, "
 _
                    & "T_ProduitEnseigne.CodeProduitEnseigne " _
                    & " FROM T_Formateur INNER JOIN T_ProduitEnseigne " _
                    & "ON T_Formateur.CodeFormateur = T_ProduitEnseigne.CodeFormateur "

Public Const cstFormateursDispo As String = "SELECT DISTINCT T_Formateur.CodeFormateur, " _
                    & "[Civilite] & "" "" & [NomFormateur] & "" "" & [PrenomFormateur] AS Formateur, "
 _
                    & "T_ProduitEnseigne.CodeNiveau, T_ProduitEnseigne.CodeProduitEnseigne,  " _
                    & "T_Planning.DateStage " _
                    & "FROM ((T_Formateur INNER JOIN T_ProduitEnseigne " _
                    & "ON T_Formateur.CodeFormateur = T_ProduitEnseigne.CodeFormateur) " _
                    & "INNER JOIN T_Stages ON T_Formateur.CodeFormateur = T_Stages.CodeFormateur) " _
                    & "INNER JOIN T_Planning ON T_Stages.CodeStage = T_Planning.CodeStage"

' Récupération des Salles
Public Const cstSalles As String = "SELECT * FROM  T_SalleFormation"

' Récupérations des salles réservées
Public Const cstSallesReservees As String = "SELECT T_Stages.CodeSalleFormation, T_Planning.DateStage "
 _
                    & "FROM T_Stages INNER JOIN T_Planning " _
                    & "ON T_Stages.CodeStage = T_Planning.CodeStage"


' Déclarations pour réservation salle
Public strSalle As String
Public varDateCliquee
Public lngCodeSalleFormation As Long, lngNiveau As Long
Public lngCodeCatalogue As Long, lngCodeStage As Long, lngCodeProduit As Long

' Divers
Public Const cstDVP As String = "DEVELOPPEZ.COM - Gestion Planning"
Public intCompteur As Integer, intDureeStage As Integer
Public intReponse As Integer, intDefilementSalle As Integer
Public boolReservationPossible As Boolean, boolFermer As Boolean

' Gestion du planning
Public varDateTraitee, varDateDebutPlanning, varDateFinPlanning
Public lngCouleurfond As Long, lngCompteurRecord As Long, lngRecordDepart As Long, lngNbRecords As Long
Public intCompteurPave As Integer, intNumLigne As Integer
Public boolEOF As Boolean
Public strSqlWherePlanning As String, strNomPave As String, strDureeStage As String
Public strNomSallePrecedent As String, strNomSalleEnCours As String, strSallesConcernees As String
```

V - CONCLUSION

Voici donc présentés les différents éléments qui vous permettront de créer et de gérer la représentation graphique des données.
Il est bien entendu que j'ai suivi un process et que bien d'autres moyens peuvent être utilisés.
L'objet de cet article était simplement de vous mettre le pied à l'étrier. Il nous restera à voir comment gérer les plages réservées (Déplacement, suppression, changement de dates ...). Ce sera l'objet de la seconde partie de cet article.

VI - TELECHARGEMENT

Pour vous permettre de matérialiser les concepts décrits ci-dessous vous pouvez télécharger la base exemple.
Base **Gestion de Planning**

VII - REMERCIEMENTS

Je voudrais remercier l'ensemble de l'équipe DVP qui fait un travail énorme qui a fait de **Développez.com** ce qu'il est aujourd'hui et qui nous tire toujours vers le haut.
Merci également à **Pierre** qui m'a aidé à débuguer un problème difficile, ainsi qu'à **Dolphy35** pour sa relecture attentive et pour le contrôle de la base exemple.
Enfin, un remerciement particulier à **Lou Pitchoun** pour son soutien durant toute la préparation de l'article.