

Suppression, mises à jour, impression des données d'un planning - Partie 2



par Jean BALLAT ([Espace perso de jeannot45](#))

Date de publication : 30/06/08

Dernière mise à jour :

Cet article continue la progression de la **gestion de planning (partie 1)**
On y retrouvera la description des manipulations complémentaires à la gestion de nos données (suppression, mises à jour diverses, notification (envoi de mail), impression et double affichage des données.
Il a été coécrit avec **Morgan Billy (Dolphy35)** pour la partie Notification

I - INTRODUCTION.....	3
II - LE CAHIER DES CHARGES.....	3
II-A - Contenu du Cahier des Charges.....	3
II-B - Présentation des modifications.....	4
III - LE DOUBLE AFFICHAGE.....	5
III-A - Présentation.....	5
III-B - Le Code VBA associé.....	5
IV - LA MANIPULATION DES DONNEES.....	8
IV-A - Présentation.....	8
IV-B - Les contrôles dans le formulaire.....	12
IV-C - Les Boutons du Formulaire.....	12
IV-C-1 - Supprimer (2).....	12
IV-C-2 - Salle (3).....	13
IV-C-3 - Date (4).....	14
IV-C-4 - Formateur (5).....	16
V - IMPRIMER LES DONNEES.....	18
V-A - Présentation.....	18
V-B - Principe du Planning.....	19
V-C - Le sous-état SE_DetailFormation.....	20
V-D - Le sous-état SE_Planning.....	21
VI - LA NOTIFICATION.....	25
VI-A - Présentation.....	25
VI-A-1 - Le Formulaire.....	25
VI-A-2 - Fonctionnement du formulaire.....	26
VI-A-3 - Structure du formulaire.....	26
VI-A-3-a - Description des contrôles.....	26
VI-A-3-b - Les propriétés des contrôles.....	27
VI-A-3-b-i - La zone de liste déroulante : IstSemaine.....	27
VI-A-3-b-ii - Zones de Texte : txtDu et txtAu.....	27
VI-A-3-b-iii - Zone de liste : IsFormateur.....	27
VI-B - Le Code VBA dans le formulaire.....	27
VII - CONCLUSION.....	30
VIII - TELECHARGEMENT.....	30
IX - REMERCIEMENTS.....	30

I - INTRODUCTION

Nous avons abordé lors de la partie 1, la génération du planning et la création de tous les formulaires nécessaires à la mise en place de celui-ci.

Dans cette seconde phase, nous allons manipuler nos données, c'est à dire les mettre à jour, les supprimer, les imprimer.

Comme dans la partie précédente, nous devons bien sûr nous conformer aux exigences du Cahier des Charges.

Cet article se décomposera en plusieurs étapes :

- Le Cahier des Charges.
- Le Double affichage (Formateurs/Salles).
- La Manipulation des données.
- L'Impression.
- La Notification (envoi de mail).

On retrouvera la base exemple GestionPlanningV2 dans la Section "Téléchargement"

II - LE CAHIER DES CHARGES

II-A - Contenu du Cahier des Charges

Le double affichage :

Afin de rendre l'application plus conviviale, il sera possible d'afficher soit le planning des salles, soit le planning des formateurs.

Lorsque l'affichage sera du type "Formateur", une sécurité sera prévue en cas de clic sur une case du planning, empêchant tout utilisateur d'intervenir sur celui-ci.

Lorsque l'affichage sera de type "Salles", les cases du planning seront sensibles au clic et permettront alors de réserver (voir [Gestion planning partie 1](#)) ou de modifier une session.

La manipulation des données :

On entendra par manipulation des données :

- Supprimer une formation.
- Changer de formateur.
- Changer de salle de formation.
- Changer la date de la formation.

L'impression :

Celle-ci imprimera la partie graphique du planning et le détail des formations. Elle dépendra du type d'affichage.

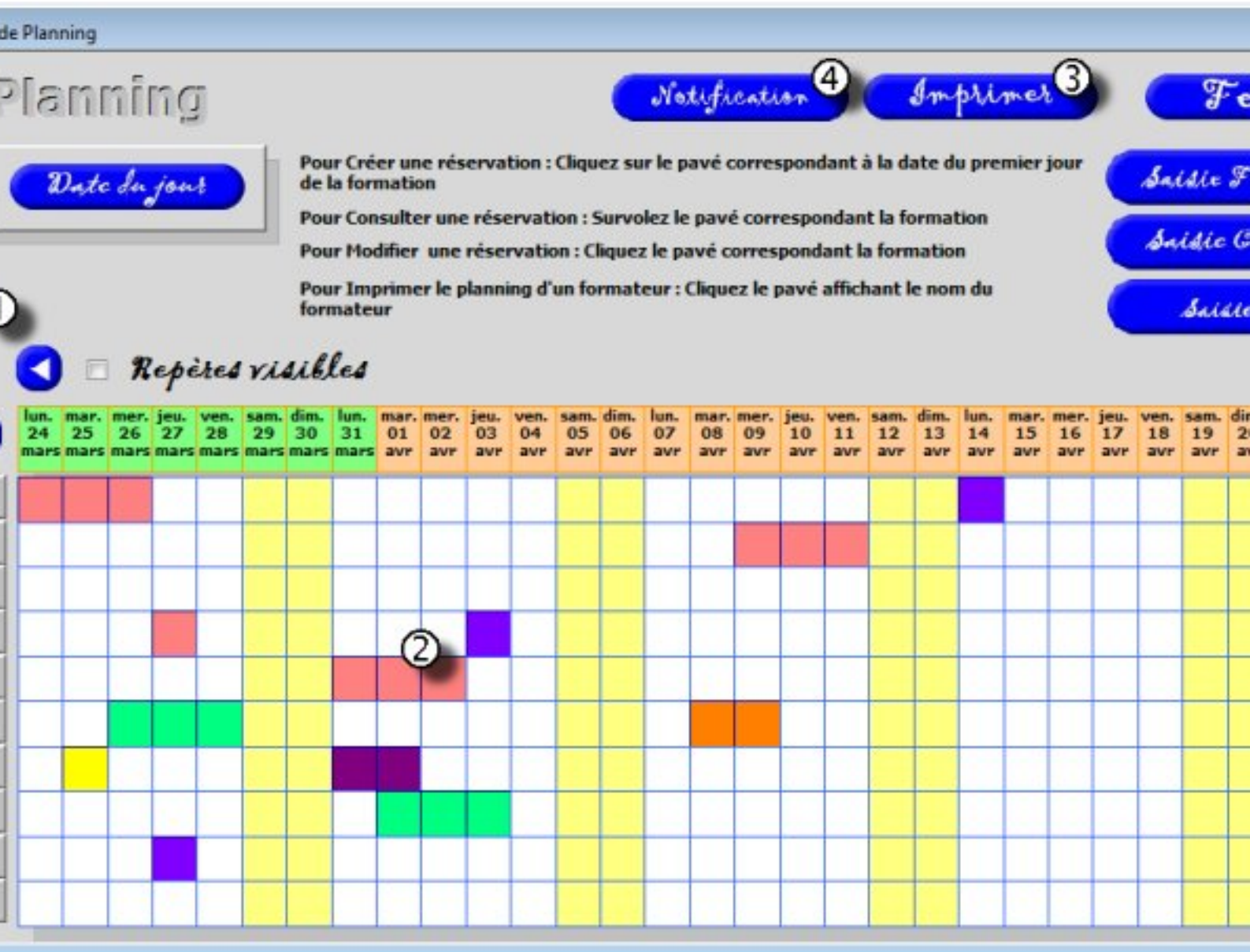
- Affichage Formateurs : Impression lancée en cliquant sur le nom du formateur.
- Affichage Salles : Impression globale de la période affichée. Impression lancée à partir du bouton "Imprimer".

La notification :

On pourra sélectionner une période libre ou une semaine particulière et envoyer un mail aux formateurs disposant d'une adresse mail.

Le mail s'affichera afin de donner la possibilité d'ajouter un texte particulier.

II-B - Présentation des modifications

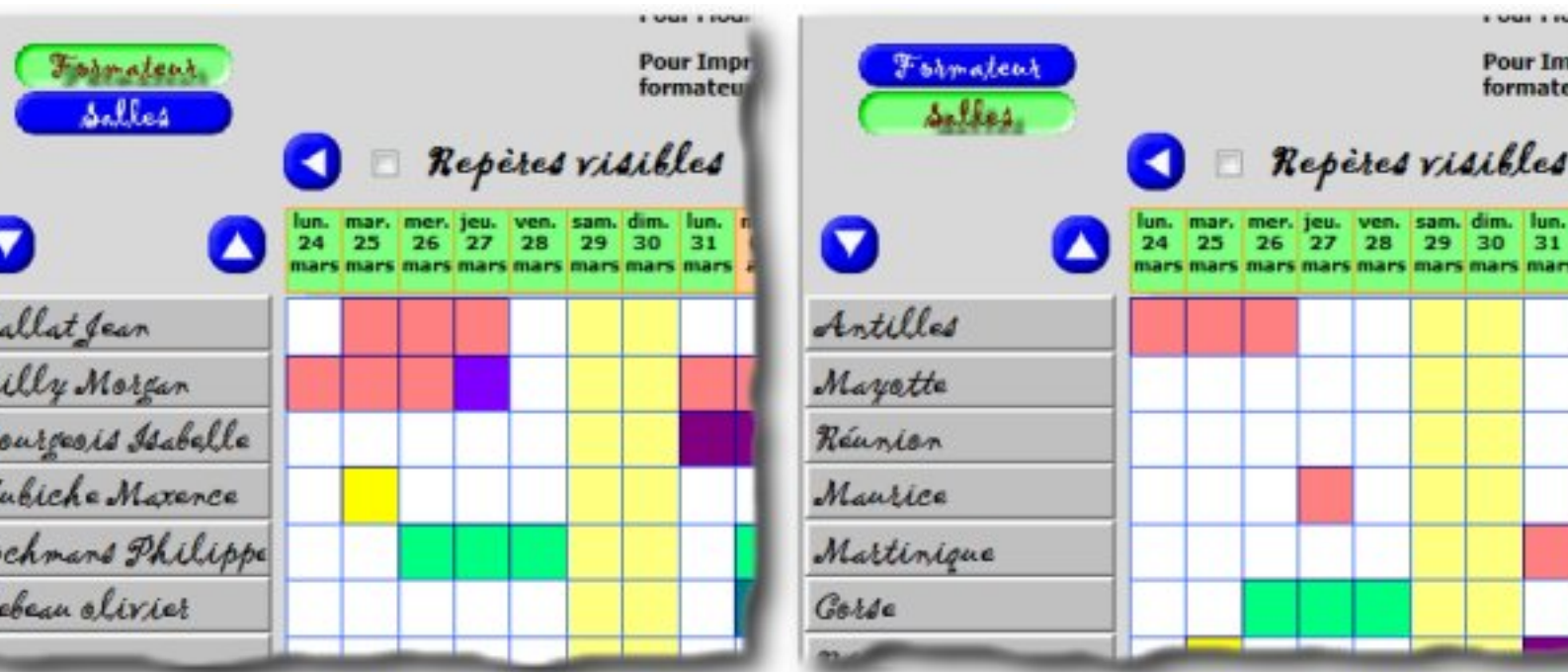


Chaque nouveauté a été matérialisée par un numéro sur l'image.

- 1) Simulation des boutons "Bascule" avec l'animation décrite dans la gestion de planning (**partie 1**).
- 2) Modification des données en cliquant sur une des cases occupées du planning.(Type d'affichage : Salles)
- 3) Impression globale du planning des salles.
- 4) La notification auprès des formateurs disposant d'une adresse mail.

III - LE DOUBLE AFFICHAGE

III-A - Présentation



Par le jeu de boutons "Bascule" personnalisés, on affiche soit le planning des formateurs soit le planning des salles. On remarquera en mode "Création" sur le formulaire F_Planning, l'ajout de zones texte ("txtCode1"... "txtCode10") qui recevront soit le CodeFormateur, soit le CodeSalleFormation. La récupération de cette information sera importante pour l'impression du planning du formateur.

On aurait pu utiliser cette méthode dans la version 1 du planning pour récupérer également le nom de la salle de formation sélectionnée.

III-B - Le Code VBA associé

Il nous faut donc savoir quel bouton a été cliqué par l'utilisateur. Pour cela, j'ai déclaré une variable **boolAffichePlanningSalle** de portée "**Public**" et de type booléen. Par défaut, à l'ouverture du formulaire F_Planning, cette variable est positionnée sur "**True**".

En cliquant sur btnPlanningFormateurS et btnPlanningSalleS on affectera à celle-ci soit la valeur **False** soit la valeur **True**.

Voici le code associé au bouton btnPlanningSalleS sur l'évènement "sur souris relâchée":

```
Private Sub btnPlanningSalleS_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single)
    ' Désactive le bouton btnPlanningSalle
    Call BasculerBouton("btnPlanningFormateurC", "btnPlanningFormateurR")

    ' initialisation des variables
    boolAffichePlanningSalle = True
    lngRecordDepart = 0
    lngCouleurfond = 8454016
    lngCodeFormateur = 0
    intDefilementSalle = 0
    boolEOF = False

    ' Mise en place du planning
```



```
RecuperationEntetesLignes
GenerationPlanningSalles
End Sub
```



On remarquera l'appel à deux routines :

- "RecuperationEntetesLignes" qui affiche soit les salles soit les formateurs.
- "GenerationPlanningSalles" qui est la routine déjà présentée dans la partie 1

Voici le code associé au bouton btnPlanningFormateurS sur l'évènement "sur souris relâchée":

```
Private Sub btnPlanningFormateurS_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single)
    ' Désactive le bouton btnPlanningSalle
    Call BasculerBouton("btnPlanningSalleC", "btnPlanningSalleR")

    ' Initialisation des variables
    boolAffichePlanningSalle = False
    lngRecordDepart = 0
    lngCouleurfond = 8454016
    intDefilementSalle = 0
    boolEOF = False

    ' Mise en place du planning
    RecuperationEntetesLignes
    GenerationPlanningFormateurs
End Sub
```



On remarquera l'appel à deux routines :

- "RecuperationEntetesLignes" qui affiche soit les salles soit les formateurs.
- "GenerationPlanningFormateurs". J'ai préféré pour plus de clarté, réécrire la routine

plutôt que de combiner les instructions dans la routine "GenerationPlanningSalles"

L'objet de cette routine est triple :

- Récupérer les infos pour renseigner les étiquettes sur le côté gauche du planning.
- Récupérer les codes Formateurs pour renseigner les contrôles invisibles dans le formulaire (Ceux-ci seront nécessaires pour l'impression du planning du formateur).
- Créer la chaîne de critères utilisée dans l'extraction des données du planning. Voyons cette routine.

```
Sub RecuperationEntetesLignes()
    ' Test du type d'affichage sélectionné
    If boolAffichePlanningSalle = True Then
        ' Réinitialisation des variables
        intCompteur = 1
        strSallesConcernees = ""

        ' Récupération du jeu d'enregistrements
        Set rsSalles = CurrentDb.OpenRecordset(cstSalles)
        ' Relecture du jeu de rsSalles pour créer la chaîne qui permettra l'extraction des données correspondantes dans rsPlanning
        With rsSalles
            ' Retour au début du jeu d'enregistrement
            .MoveFirst
            ' Saute au premier enregistrement concerné

            ' Rappel : lngRecordDepart permet de repositionner le pointeur suivant le nombre de clics sur les boutons de défilement vers le haut ou vers le bas
            .Move lngRecordDepart
            ' Boucle jusqu'à la fin du jeu d'enregistrements
            Do While Not .EOF
                Me.Controls("lblNomSalle" & intCompteur).Caption = .Fields(1)
                Me.Controls("txtCode" & intCompteur) = .Fields(0)
                ' Génération de la chaîne de critères qui sera utilisée avec l'opérateur IN
                strSallesConcernees = strSallesConcernees & "," & Chr(34) & .Fields(1) & Chr(34)
            Loop
        End With
    End If
End Sub
```

```

        intCompteur = intCompteur + 1
        ' Valeur limite pour sortir de la boucle
        If intCompteur > 10 Then
            Exit Do
        Else
            .MoveNext
        End If
    Loop
    ' Initialisation 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)
Else
    ' Réinitialisation des variables
    intCompteur = 1
    strFormateursConcernes = ""

    ' Récupération du jeu d'enregistrements
    Set rsFormateurs = CurrentDb.OpenRecordset(cstFormateurs)
    ' Relecture du jeu de rsFormateurs pour créer la chaîne qui permettra
    ' l'extraction des données correspondantes dans rsPlanning
    With rsFormateurs
        ' Retour au début du jeu d'enregistrement
        .MoveFirst
        ' Saute au premier enregistrement concerné

    ' Rappel : lngRecordDepart permet de repositionner le pointeur suivant le nombre de clics sur les
    ' boutons de défilement vers le haut ou vers le bas
        .Move lngRecordDepart
        ' Boucle jusqu'à la fin du jeu d'enregistrements
        For intCompteur = 1 To 10
            If .EOF Then
                ' Initialisation de la variable de fin de fichier
                boolEOF = True
                Me.Controls("lblNomSalle" & intCompteur).Caption = " "
                Me.Controls("txtCode" & intCompteur) = 0
            Else
                Me.Controls("lblNomSalle" & intCompteur).Caption = .Fields(1)
                Me.Controls("txtCode" & intCompteur) = .Fields(0)
                ' Génération de la chaîne de critères qui sera utilisée avec l'opérateur IN
                strFormateursConcernes = strFormateursConcernes & "," & .Fields(0)
                .MoveNext
            End If
        Next
    End With

    ' enlève la virgule
    strFormateursConcernes = Right(strFormateursConcernes, Len(strFormateursConcernes) - 1)
End If
End Sub

```



On remarque que dans la boucle, on renseigne les champs "lblNomSalle" avec les noms des salles ou des formateurs et "txtCode" avec le CodeFormateur ou le CodeSalleFormation. Le codeFormateur servira lors de l'impression du planning du formateur.

Rien d'autre de particulier dans cette procédure, si ce n'est que le test de la variable boolAffichePlanningSalle dirige tout.


IV - LA MANIPULATION DES DONNEES

IV-A - Présentation

La manipulation des données n'est accessible qu'à partir du mode d'affichage "Salles".

Ainsi en cliquant sur une plage colorée, l'utilisateur se verra avertir par une boîte de dialogue que la zone est déjà occupée et que s'il le souhaite, il pourra intervenir sur la session en cours.



 Cependant, si l'utilisateur clique sur une zone du planning en mode "Formateurs", une boîte de dialogue proposera de basculer en mode "Salles" afin de pouvoir faire des modifications ou une réservation.
Ci-dessous le code VBA associé.

```
Sub Reservation()
    ' teste le type d'affichage
    If boolAffichePlanningSalle = False Then

        intReponse = MsgBox("La réservation ou la modification n'est possible qu'en mode affichage ""Planning Salles""",
        -
        "Souhaitez vous basculer dans ce mode d'affichage", vbQuestion + vbYesNo,
        cstDVP)
        ' Test de la réponse de l'utilisateur
    End If
End Sub
```



```
If intReponse = vbYes Then
    ' Active le bouton "Salles" au lieu du bouton "Formateurs"
    Call BasculerBouton("btnPlanningFormateurC", "btnPlanningFormateurR")
    If btnPlanningSalleS.Visible = True Then
        Call BasculerBouton("btnPlanningSalleS", "btnPlanningSalleC")
    Else
        Call BasculerBouton("btnPlanningSalleR", "btnPlanningSalleC")
    End If
    ' Réinitialisation de la variable booléenne
    boolAffichePlanningSalle = True
    ' Régénération du planning
    RecuperationEntetesLignes
    GenerationPlanningSalles
End If
Else
    ' 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 If
End Sub
```

L'affichage est donc repositionné en mode "Salles". Un clic sur une plage blanche génère alors la création d'une nouvelle réservation.

C'est en cliquant sur une des cases réservées d'une salle que le process de modification se met en route. Ce clic entraîne l'affichage du formulaire ci dessous.

Ce formulaire se décompose en deux parties :

- à gauche : les commandes.
- dans l'ardoise, les infos disponibles.

Par infos disponibles, j'entends les infos affichées en fonction du bouton choisi :

- bouton "Salle" : Liste les salles disponibles pour la période réservée de la formation.
- bouton "Date" : Affiche un calendrier afin de choisir le début de la nouvelle période.
- bouton "Formateur" : Liste les formateurs compétents et disponibles pour la période de la formation.

Lors de l'ouverture du formulaire, une procédure est lancée afin de reconnaître la formation concernée par la modification.

```
Private Sub Form_Open(Cancel As Integer)
    Dim rsFormationConcernee As DAO.Recordset
    Dim strSqlFormationConcernee As String

    Set frmActif = Me

    ' Charge les images
    btnFermerR.Picture = CurrentProject.Path & "\image\btnFermerR.jpg"
    btnFermerS.Picture = CurrentProject.Path & "\image\btnFermerS.jpg"
    btnFermerC.Picture = CurrentProject.Path & "\image\btnFermerC.jpg"
    btnSupprimerR.Picture = CurrentProject.Path & "\image\btnSupprimerR.jpg"
    btnSupprimerS.Picture = CurrentProject.Path & "\image\btnSupprimerS.jpg"
    btnSupprimerC.Picture = CurrentProject.Path & "\image\btnSupprimerC.jpg"
    btnModifSalleR.Picture = CurrentProject.Path & "\image\btnModifSalleR.jpg"
```

```

btnModifSalleS.Picture = CurrentProject.Path & "\image\btnModifSalleS.jpg"
btnModifSalleC.Picture = CurrentProject.Path & "\image\btnModifSalleC.jpg"
btnModifDateR.Picture = CurrentProject.Path & "\image\btnModifDateR.jpg"
btnModifDateS.Picture = CurrentProject.Path & "\image\btnModifDateS.jpg"
btnModifDateC.Picture = CurrentProject.Path & "\image\btnModifDateC.jpg"
btnModifFormateurR.Picture = CurrentProject.Path & "\image\btnModifFormateurR.jpg"
btnModifFormateurS.Picture = CurrentProject.Path & "\image\btnModifFormateurS.jpg"
btnModifFormateurC.Picture = CurrentProject.Path & "\image\btnModifFormateurC.jpg"

' récupération des informations de la formation

strSqlFormationConcernee = "SELECT T_Stages.CodeStage, T_Stages.DateStageDebut, T_Catalogue.DureeStage, " _
-
& "T_NiveauFormation.LibelleNiveau, T_Catalogue.IntituleStage, " _
& "[NomFormateur] & " " " & [PrenomFormateur] AS Formateur, T_Produits.CodeProduit, " _
& "T_NiveauFormation.CodeNiveau " _
& "FROM T_NiveauFormation " _
& "INNER JOIN (T_Produits " _
& "INNER JOIN (T_Catalogue " _
& "INNER JOIN (T_Formateur " _
& "INNER JOIN (T_SalleFormation " _
& "INNER JOIN T_Stages " _
& "ON T_SalleFormation.CodeSalleFormation = T_Stages.CodeSalleFormation) " _
& "ON T_Formateur.CodeFormateur = T_Stages.CodeFormateur) " _
-
& "ON T_Catalogue.CodeCatalogue = T_Stages.CodeCatalogue) " _
-
& "ON T_Produits.CodeProduit = T_Catalogue.CodeProduit) " _
& "ON T_NiveauFormation.CodeNiveau = T_Catalogue.CodeNiveau " _
& "WHERE T_Stages.CodeStage = " & lngCodeStage

' Initialisation du jeu d'enregistrements
Set rsFormationConcernee = CurrentDb.OpenRecordset(strSqlFormationConcernee)

With rsFormationConcernee
' récupération de la durée de la formation
bytDureeFormation = .Fields(2)
' Récupération du produit concerné
lngCodeProduit = .Fields(6)
' Récupération du Niveau de formation
lngNiveau = .Fields(7)
' préparation du message affiché dans l'ardoise
lblMessage.Caption = "Formation n° : " & lngCodeStage & vbCrLf _
& "Intitule : " & .Fields(4) & vbCrLf _
& "Durée : " & IIf(bytDureeFormation = 1, bytDureeFormation _
& " jour", bytDureeFormation & " jours") & vbCrLf _
& "Début le : " & .Fields(1) & vbCrLf _
& "Formateur : " & .Fields(5)

varDateDebutFormation = .Fields(1)
varDateFinFormation = DateAdd("d", bytDureeFormation - 1, varDateDebutFormation)
End With
End Sub

```

Etant donné que les boutons "Salle", "Date", "Formateur" sont encadrés, il faut également envisager une procédure pour l'animation des boutons. Celle-ci est affectée à l'évènement "sur souris déplacée" du contrôle "objEncadrementBoutons". On remarquera que cette procédure fait appel à la procédure événementielle "sur souris déplacée" de la zone Détail.

```

Private Sub objEncadrementBoutons_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)
' appel de la routine permettant de gérer l'animation des boutons
Call Détail_MouseMove(Button, Shift, X, Y)
End Sub

```

```


Private Sub Détail_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)
    ' Réinitialisation des images
    btnFermerR.Visible = True
    btnFermerS.Visible = False
    btnFermerC.Visible = False
    btnSupprimerR.Visible = True
    btnSupprimerS.Visible = False
    btnSupprimerC.Visible = False
    btnModifSalleR.Visible = True
    btnModifSalleS.Visible = False
    btnModifSalleC.Visible = False
    btnModifDateR.Visible = True
    btnModifDateS.Visible = False
    btnModifDateC.Visible = False
    btnModifFormateurR.Visible = True
    btnModifFormateurS.Visible = False
    btnModifFormateurC.Visible = False
End Sub

```

IV-B - Les contrôles dans le formulaire

Hormis les boutons donc l'explication sur l'animation peut-être consultée dans la **partie 1 de l'article**, ce formulaire contient dans l'ardoise, 3 autres contrôles superposés (1).

- Une étiquette (lblMessage) qui affiche les infos sur la formation en cours de modification.
- Un contrôle calendrier (objCalendrier) qui permettra de sélectionner une nouvelle date (4).
- Une zone de liste (lstElementDispo) qui permettra d'afficher soit les salles (3) disponibles soit les formateurs (5) compétents et disponibles.

 Les numéros entre parenthèses indiquent le bouton utilisé (cf image du formulaire)

IV-C - Les Boutons du Formulaire

IV-C-1 - Supprimer (2)

Le rôle de ce bouton est de supprimer la formation sélectionnée. Lors de la suppression de la formation, tous les enregistrements attachés de la table T_Planning seront également supprimés. (propriété de l'intégrité référentielle : Effacer en cascade les enregistrements correspondants).

```

Private Sub btnSupprimerS_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single)
    ' Déclaration variable locale
    Dim strDeleteStage As String

    ' Initialisation de la syntaxe SQL nécessaire à la suppression
    strDeleteStage = "DELETE CodeStage FROM T_Stages WHERE CodeStage = " & lngCodeStage

    ' annule l'affichage des messages système
    DoCmd.SetWarnings False
    ' interrogation de l'utilisateur
    intReponse = MsgBox("Vous allez supprimer définitivement une formation !" & _
        & vbCrLf & "Souhaitez-vous continuer ?", vbQuestion + vbYesNo, cstDVP)

    ' Traitement de la réponse de l'utilisateur
    ' Confirmation de la suppression
    If intReponse = vbYes Then
        ' Exécute la requête de suppression
        DoCmd.RunSQL strDeleteStage
    Else
        ' Annule la suppression des enregistrements
        MsgBox "La demande de suppression de la formation a été annulée !", vbInformation, cstDVP
    End If
End Sub

```

```

' Réactivation des messages système
DoCmd.SetWarnings True

Set frmActif = Forms("F_Planning").Form

' Fermeture du formulaire
DoCmd.Close

' Redessine le planning
GenerationPlanningSalles

End Sub

```

IV-C-2 - Salle (3)

Le changement de salle nous amène à proposer à l'utilisateur une nouvelle salle à la condition qu'elle soit disponible pour la période demandée.

La procédure affiche alors toutes les salles disponibles pour la période de la formation concernée par la modification. Pour faire une modification de salle, deux procédures seront nécessaires. Une première pour récupérer les salles disponibles qui se lance dès que je relâche le bouton et une seconde qui relance le planning après avoir choisi une salle dans la liste (sur l'évènement "Après MAJ" de la liste lstElementDispo). Ci dessous le code de la procédure du bouton btnModifSalles sur l'évènement "sur souris relâchée".

```

Private Sub btnModifSalles_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single)
    Dim strSqlSallesReservees As String, strSallesOccupees As String

    ' Préparation de la récupération des codes salles occupées

    strSqlSallesReservees = "SELECT T_Stages.CodeStage, T_SalleFormation.CodeSalleFormation, T_Stages.DateStageDebut
    -
                                & "FROM T_SalleFormation INNER JOIN T_Stages " _
    & "ON T_SalleFormation.CodeSalleFormation = T_Stages.CodeSalleFormation " _
                                & "WHERE T_Stages.DateStageDebut Between #" &
    Format(varDateDebutFormation, "mm/dd/yy") _
                                & "# And #" & Format(varDateFinFormation, "mm/dd/yy") & "#"

    ' Récupération du jeu d'enregistrements
    Set rsSallesReservees = CurrentDb.OpenRecordset(strSqlSallesReservees)

    ' Création de la chaîne des salles occupées (qui seront exclues de l'extraction)
    With rsSallesReservees
        Do While Not .EOF
            strSallesOccupees = strSallesOccupees & "," & .Fields(1)
            .MoveNext
        Loop
    End With

    ' Suppression de la virgule en fin de chaîne
    strSallesOccupees = Right(strSallesOccupees, Len(strSallesOccupees) - 1)

    ' initialisation de la variable qui sera la source de la zone de liste
    strSqlSallesReservees = "SELECT * FROM T_SalleFormation WHERE CodeSalleFormation NOT IN (" &
    strSallesOccupees & ")"

    ' Alimentation de la liste lstElementDispo
    With lstElementDispo
        .RowSourceType = "Table/Query"
        .RowSource = strSqlSallesReservees
    End With

    ' Changement de l'intitulé de l'ardoise
    lblFormationModif1.Caption = "Choisir une Salle"
    lblFormationModif2.Caption = "Choisir une Salle"

    ' Masque le message de la formation concernée et le calendrier pour afficher la liste des salles disponibles

```



```
lblMessage.Visible = False
objCalendrier.Visible = False
lstElementDispo.Visible = True

' Permet d'utiliser le même objet pour deux sources différentes. (Salles ou Formateurs)
boolModifFormateur = False
```

End Sub

Voici le code associé à l'évènement "Après MAJ" de la liste lstElementDispo :

```
Private Sub lstElementDispo_AfterUpdate()
    Dim strElementMAJ As String

    ' Désactivation des messages système
    DoCmd.SetWarnings False

    ' test sur le type d'info à mettre à jour
    ' (boolModifFormateur = False -> Modif Salles)
    ' (boolModifFormateur = True -> Modif Formateur)
    If boolModifFormateur = False Then
        ' Mise à jour du CodeSalleFormation
        strElementMAJ = "CodeSalleFormation"
    Else
        ' Mise à jour du CodeFormateur
        strElementMAJ = "CodeFormateur"
    End If

    ' Préparation de la requete MAJ
    strSQLMajStage = "UPDATE T_Stages SET " & strElementMAJ & " = " & lstElementDispo & _
        " WHERE CodeStage = " & lngCodeStage

    DoCmd.RunSQL strSQLMajStage

    ' réactivation des messages système
    DoCmd.SetWarnings True

    ' redessine le planning
    GenerationPlanningSalles
    ' Ferme le formulaire de modification de la formation
    DoCmd.Close acForm, "F_ModifReservation"
End Sub
```

On remarquera que cette procédure servira pour les deux modifications (salle ou formateur) et que la mise à jour porte uniquement sur la modification du code (Formateur ou Salle de formation).

IV-C-3 - Date (4)

Pour modifier la date, un simple objet calendrier facilitera la tâche de l'utilisateur. Il apparaîtra également dans l'ardoise.

L'affichage de celui-ci est donc attaché au bouton btnModifDateS sur l'évènement "sur souris relâchée".

```
Private Sub btnModifDateS_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single)
    ' Changement de l'intitulé de l'ardoise
    lblFormationModif1.Caption = "Choisir une nouvelle Date"
    lblFormationModif2.Caption = "Choisir une nouvelle Date"

    ' Affiche le Calendrier
    objCalendrier.Visible = True
    lblMessage.Visible = False
    lstElementDispo.Visible = False
    boolModifDate = True
End Sub
```

Après avoir choisi une nouvelle date, un contrôle de validité est effectué et le planning est redessiné. Voici le code associé à l'objet objCalendrier sur l'évènement "sur Clic".

```
Private Sub objCalendrier_Click()
    ' Récupération de la date cliquée
    varDateDebutFormation = objCalendrier.Value

    ' Contrôle de validité de la date et de la disponibilité de la salle
    boolModifPossible = ControleReservation(varDateDebutFormation, bytDureeFormation)

    If boolModifPossible = True Then
        ' Contrôle de disponibilité du formateur
        If ControleDispoFormateur(varDateDebutFormation, bytDureeFormation) = True Then
            ' Désactivation des messages système
            DoCmd.SetWarnings False

            ' Mise à jour de la formation
            strSQLMajStage = "UPDATE T_Stages SET DateStageDebut = #" &
Format(varDateDebutFormation, "mm/dd/yy") & _
"# WHERE CodeStage = " & lngCodeStage
            DoCmd.RunSQL strSQLMajStage

            ' Mise à jour des dates du planning
            For intCompteur = 0 To bytDureeFormation - 1
                varDateFinFormation = Format(DateAdd("d", intCompteur, varDateDebutFormation), "mm/dd/yy")
                strSQLMajStage = "UPDATE T_Planning SET DateStage = #" & varDateFinFormation & _
"# WHERE CodeStage = " & lngCodeStage & " AND Quantieme = " & intCompteur + 1
                DoCmd.RunSQL strSQLMajStage
            Next

            ' Réactivation des messages système
            DoCmd.SetWarnings True

            ' redessine le planning
            GenerationPlanningSalles

            ' Ferme le formulaire de modification de la formation
            DoCmd.Close acForm, "F_ModifReservation"
        End If
    End If
End Sub
```

On remarquera dans la procédure, l'utilisation d'une fonction personnalisée dont voici le détail ci-dessous :

```
Public Function ControleReservation(varDateChoisie, lngDureeFormation) As Boolean
    Dim varDateFinReservation, varDateDebutReservation, strSqlWhereReservation As String

    ' Contrôle date début Formation
    If Weekday(varDateChoisie) = vbFriday And lngDureeFormation > 1 Or _
Weekday(varDateChoisie) = vbThursday And lngDureeFormation > 2 Then

        MsgBox "La formation est tronquée par une fin de semaine" & vbCrLf & "Veuillez revoir votre choix", vbCritical,
cstDVP
        boolReservationPossible = False
    Else
        ' Définition des bornes de la formation
        varDateDebutReservation = Format(varDateChoisie, "mm/dd/yy")
        varDateFinReservation = Format(DateAdd("d", lngDureeFormation - 1, varDateChoisie), "mm/dd/yy")

        ' Test pour vérification du type de contrôle (pour création ou pour modification)
        strSqlWhereReservation = " WHERE T_Stages.CodeSalleFormation = " & lngCodeSalleFormation _
& " AND DateStage BETWEEN #" & varDateDebutReservation _
& "# AND #" & varDateFinReservation & "#"

        ' Récupération des données de la réservation
```

```

Set rsSallesReservees = CurrentDb.OpenRecordset(cstSallesReservees & strSqlWhereReservation)

With rsSallesReservees
    If .RecordCount > 0 Then
        ' test de contrôle qu'une modification est en cours
        If boolModifDate = False Then
            Do While Not .EOF
                If CDate(varDateFinReservation) >= Format(.Fields(1), "mm/dd/yy") Then
                    MsgBox "la salle est déjà occupée par une autre formation", vbCritical, cstDVP
                    boolReservationPossible = False
                    Exit Do
                Else
                    .MoveNext
                End If
            Loop
        Else
            ' Désactivation du mode Modification
            boolModifDate = False
            ' Permet la modification demandée
            boolReservationPossible = True
        End If
    Else
        boolReservationPossible = True
    End If
End With
End If

' Affectation de la valeur à la fonction
If boolReservationPossible = False Then
    ControleReservation = False
Else
    ControleReservation = True
End If
End Function

```

IV-C-4 - Formateur (5)

L'intérêt de cette modification est de pouvoir affecter un autre formateur à la formation sélectionnée. Cependant, le formateur choisi devra avoir la compétence et être disponible durant toute la session de la formation. Comme pour le changement de salle, deux procédures seront nécessaires pour terminer la tâche.

- Afficher les formateurs compétents et disponibles.
- Mettre à jour la formation en fonction du nouveau formateur.

Dans un premier temps, découvrons le code associé au bouton btnModifFormateurS sur l'évènement "sur souris relâchée" :

```

Private Sub btnModifFormateurS_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single)
    Dim strSqlFormateursReserves As String, strFormateursIndispos As String
    Dim rsFormateurReserves As DAO.Recordset

    ' Changement de l'intitulé de l'ardoise
    lblFormationModif1.Caption = "Choisir un Formateur"
    lblFormationModif2.Caption = "Choisir un Formateur"

    ' Préparation de la récupération des formateurs

    strSqlFormateursReserves = "SELECT T_Formateur.CodeFormateur, T_Catalogue.CodeNiveau, T_Catalogue.CodeProduit, " & _
    "T_Stages.DateStageDebut, T_Catalogue.DureeStage " & _
    "FROM T_Formateur INNER JOIN (T_Catalogue INNER JOIN T_Stages " & _
    "ON T_Catalogue.CodeCatalogue = T_Stages.CodeCatalogue) " & _
    "ON T_Formateur.CodeFormateur = T_Stages.CodeFormateur " & _
    "WHERE T_Stages.DateStageDebut Between #" & _
    Format(varDateDebutFormation, "mm/dd/yy") & _

```

```

                                "# And #" & Format(DateAdd("d", bytDureeFormation - 1,
varDateDebutFormation), "mm/dd/yy") & "#"

' Création du jeu d'enregistrements
Set rsFormateurReserves = CurrentDb.OpenRecordset(strSqlFormateursReserves)

With rsFormateurReserves
' Création des valeurs qui seront exclues de la liste des formateurs par l'opérateur NOT IN
If .RecordCount > 0 Then
    Do While Not .EOF
        strFormateursIndispos = strFormateursIndispos & "," & .Fields(0)
        .MoveNext
    Loop
Else
    MsgBox "Il n'y a pas de formateurs disponibles pour la date choisie" _
    & vbCrLf & "Veuillez refaire votre choix", vbInformation, cstDVP
    Exit Sub
End If
End With

' Suppression de la virgule
strFormateursIndispos = Right(strFormateursIndispos, Len(strFormateursIndispos) - 1)

' Sélection des formateurs disponibles avec la compétence dans la période
strSqlFormateursReserves = "SELECT T_Formateur.CodeFormateur, " _

& "[Civillite] & "" "" & [NomFormateur] & "" "" & [PrenomFormateur] AS Formateur " _
    & "FROM T_Formateur INNER JOIN T_ProduitEnseigne " & _
    & "ON T_Formateur.CodeFormateur = T_ProduitEnseigne.CodeFormateur " & _
    & "WHERE T_ProduitEnseigne.CodeNiveau = " & lngNiveau _
    & " AND T_ProduitEnseigne.CodeProduitEnseigne = " & lngCodeProduit & _
    & "AND T_Formateur.CodeFormateur NOT IN(" _
    & strFormateursIndispos & ") ORDER BY NomFormateur"

' Vérification du contenu
Set rsFormateurReserves = CurrentDb.OpenRecordset(strSqlFormateursReserves)
If rsFormateurReserves.RecordCount = 0 Then

MsgBox "Il n'y a pas de formateurs disponibles pour cette formation à cette date", vbInformation,
cstDVP

    Exit Sub
End If

' Masque le message de la formation concernée pour afficher la liste des formateurs disponibles
lblMessage.Visible = False
' Masque le calendrier
objCalendrier.Visible = False
' affiche la liste des formateurs compétents et dispo pour la période prévue
With lstElementDispo
    .RowSourceType = "Table/Query"
    .RowSource = strSqlFormateursReserves
    .Visible = True
End With

' Permet d'utiliser le même objet pour deux sources différentes. (Salles = False ou Formateurs = True)
boolModifFormateur = True
End Sub

```

La seconde s'exécute sur l'évènement "Après MAJ" de la liste lstElementDispo. C'est la variable "boolModifFormateur" qui permettra de modifier soit le CodeFormateur soit le CodeSalleFormation. Voici donc, pour rappel, la procédure utilisée :

```

Private Sub lstElementDispo_AfterUpdate()
    Dim strElementMAJ As String

    ' Désactivation des messages système
    DoCmd.SetWarnings False

```

```
' test sur le type d'info à mettre à jour
' (boolModifFormateur = False -> Modif Salles)
' (boolModifFormateur = True -> Modif Formateur)
If boolModifFormateur = False Then
' Mise à jour du CodeSalleFormation
strElementMAJ = "CodeSalleFormation"
Else
' Mise à jour du CodeFormateur
strElementMAJ = "CodeFormateur"
End If

' Préparation de la requete MAJ
strSQLMajStage = "UPDATE T_Stages SET " & strElementMAJ & " = " & lstElementDispo & _
                " WHERE CodeStage = " & lngCodeStage

DoCmd.RunSQL strSQLMajStage

' réactivation des messages système
DoCmd.SetWarnings True

' redessine le planning
GenerationPlanningSalles
' Ferme le formulaire de modification de la formation
DoCmd.Close acForm, "F_ModifReservation"
End Sub
```

V - IMPRIMER LES DONNEES

V-A - Présentation

Pour respecter le Cahier des Charges, nous aurons deux impressions:

- L'impression globale en partant du bouton "Imprimer" en affichage "Salles".
- L'impression du planning d'un formateur en cliquant sur le nom du Formateur.

mars 08 au 24 Planning du 24 mars 08 au 24 avr 08



Antilles

Le	IntituleStage	Niveau	Duree	Formateur
24/03/2008	Citer des tableaux et les mettre en forme	Débutant	3 jours	5 Monsieur Billy Morgan
14/04/2008	Prise en main de la messagerie	Débutant	1 jour	5 Monsieur Billy Morgan

Martinique

Le	IntituleStage	Niveau	Duree	Formateur
31/03/2008	Créer des tableaux et les mettre en forme	Débutant	3 jours	Monsieur Billy Morgan

Maurice

Le	IntituleStage	Niveau	Duree	Formateur
03/04/2008	Gérer les contacts et les réunions	Avancé	1 jour	5 Monsieur Billy Morgan

Mayotte

Le	IntituleStage	Niveau	Duree	Formateur
09/04/2008	Coder des tableaux et les mettre en forme	Débutant	3 jours	5 Monsieur Billy Morgan

Ouessant

Le	IntituleStage	Niveau	Duree	Formateur
27/03/2008	Prise en main de la messagerie	Débutant	1 jour	Monsieur Billy Morgan

IntituleStage				
Application avec VBA	Avancé	3 jours	6	Monsieur Lebeau Olivier
Traitement de texte	Débutant	2 jours	4	Monsieur Lebeau Olivier
Nettoyage du serveur	Débutant	1 jour	1	Monsieur Basset Jean

Impression Planning des Salles

V-B - Principe du Planning

L'impression d'un planning se divise en deux parties. D'une part "SE_Planning" qui représente la partie graphique (les "petites cases colorées") et d'autre part, "SE_DetailFormation" qui reprend le détail des différentes formations regroupées par salle.

Chaque partie fait l'objet d'un sous-état, les deux sous-états étant regroupés dans un état indépendant "E_Planning". A noter que les mêmes structures seront utilisées aussi bien pour l'impression des salles que pour l'impression du planning du Formateur.

[illegible]

Voyons la procédure VBA attachée à l'évènement "sur ouverture de l'état".

```

Private Sub Report_Open(Cancel As Integer)
    ' Déclaration des variables locales
    Dim strSqlImpressionSousEtat As String, strSqlWhereSousEtat As String

    ' Inititalisation de la requete source de l'état
    strSqlImpressionSousEtat = "SELECT T_Catalogue.IntituleStage, T_NiveauFormation.LibelleNiveau, " & _
    "IIf([DureeStage]>1,[DureeStage] & "" jours"",[DureeStage] & "" jour"") AS Duree, " & _
    "[Civillite] & "" "" & [NomFormateur] & "" "" & [PrenomFormateur] AS Formateur, " & _
    "T_Stages.DateStageDebut, T_SalleFormation.NomSalleFormation, " & _
    "T_Produits.NomProduit, T_Formateur.CodeFormateur " & _
    "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) " & _
    "ON T_SalleFormation.CodeSalleFormation=T_Stages.CodeSalleFormation "

    ' Inititalisation de la clause Where en fonction de la valeur de la variable
    If boolAffichePlanningSalle = True Then
        strSqlWhereSousEtat = "WHERE T_Stages.DateStageDebut Between #" & _
        & Format(varDateDebutPlanning, "mm/dd/yy") & _
        & "# And #" & Format(varDateFinPlanning, "mm/dd/yy") & "# "
    Else
        strSqlWhereSousEtat = "WHERE T_Stages.DateStageDebut Between #" & _
        & Format(varDateDebutPlanning, "mm/dd/yy") & _
        & "# And #" & Format(varDateFinPlanning, "mm/dd/yy") & _
        & "# AND T_Formateur.CodeFormateur = " & lngCodeFormateur

    End If

    ' Affectation de la source à l'état

```

```
Me.RecordSource = strSqlImpressionSousEtat & strSqlWhereSousEtat
End Sub
```


Enfin, afin d'assurer une meilleure lisibilité des lignes de l'état, j'affecte une couleur sur les lignes paires via une procédure attachée à l'évènement "sur formatage" de la section Détail.

Voyons le détail ci-dessous :

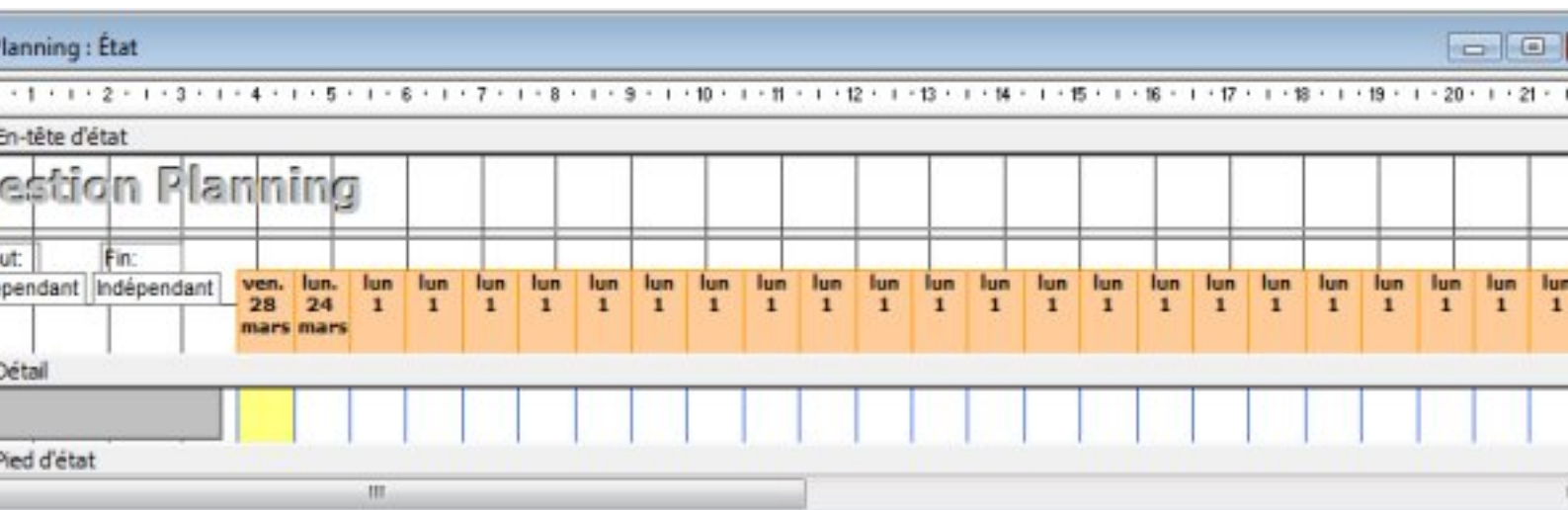
```
Private Sub Détail_Format(Cancel As Integer, FormatCount As Integer)

' Déclaration d'une variable locale statique (c'est à dire n'est pas remise à zéro tant que le processus n'est pas terminé)
Static lngNumLigne As Long
' Permet l'alternance des couleurs en fonction du type de ligne (paire ou impaire)
If lngNumLigne Mod 2 = 0 Then
    Détail.BackColor = 11206570 ' vert clair
Else
    Détail.BackColor = 16777215 ' blanc
End If

' Incrémentation du compteur de lignes
lngNumLigne = lngNumLigne + 1
End Sub
```

 Afin d'avoir de plus amples informations concernant l'affectation de plusieurs sources à un même état, je vous conseille la lecture de : **[Comment utiliser plusieurs requêtes pour un même état](#)**

V-D - Le sous-état SE_Planning



A propos de ce sous-état, deux remarques importantes :

- Il faut représenter la partie graphique. Il nous faut donc récupérer la couleur des plages de réservation.

Je récupère donc les informations nécessaires à la représentation graphique dans une table. Cette table servira aussi bien pour la version "Salles" que pour la version "Formateur". La génération de la table est lancée en cliquant soit sur le bouton "Imprimer" soit sur le nom du Formateur dont on souhaite imprimer le planning.

- Tous les contrôles de l'état sont des contrôles indépendants dont la source sera affectée au formatage de l'état.

Découvrons ci-dessous le process de la création de la table :

```
Private Sub btnImprimerS_Click()
```

```

' Déclaration des variables et des constantes de portée locale
Dim rsImprimerPlanning As DAO.Recordset, rsTypeImprimer As DAO.Recordset
Dim strSqlImprimerPlanning As String, strSqlDeleteImpressionPlanning As String,
strSqlFormateurImprime As String
Dim varDateBoucle
Dim lngCodeSalleEnCours As Long, lngCodeSallePrecedente As Long
Dim intCompteurColonnes As Integer, intCompteurLignes, strInfoBulle As String
Dim boolReservation As Boolean, boolEOFPlanning As Boolean
Const cstCouleurWeekEnd As Long = 8454143, cstCouleurVide As Long = 16777215

On Error Resume Next

' Initialisation des requetes
strSqlImprimerPlanning = "SELECT * FROM T_ImpressionPlanning"
strSqlDeleteImpressionPlanning = "DELETE CodeSalleFormation FROM T_ImpressionPlanning " & _
    "WHERE CodeSalleFormation Is Not Null"

' Test sur le type d'affichage
If boolAffichePlanningSalle = True Then
    strSqlPlanning = " Where DateStage BETWEEN #" & Format(CDate(varDateDebutPlanning), "mm/dd/yy")
    -
    & "# AND #" & Format(varDateFinPlanning, "mm/dd/yy") _
    & "# ORDER BY T_SalleFormation.CodeSalleFormation, T_Planning.DateStage"
Else
    If lngCodeFormateur = 0 Then

MsgBox "L'impression d'un formateur se fait en cliquant sur le nom du formateur concerné", vbCritical,
cstDVP

Exit Sub
End If
    strSqlPlanning = " Where DateStage BETWEEN #" & Format(CDate(varDateDebutPlanning), "mm/dd/yy")
    -
    & "# AND #" & Format(varDateFinPlanning, "mm/dd/yy") _
    & "# AND T_Formateur.CodeFormateur = " &
Controls("txtCode" & intNumLigne) _
    & " ORDER BY T_Planning.DateStage"

lngCodeFormateur = Controls("txtCode" & intNumLigne)
strSqlFormateurImprime = " Where CodeFormateur = " & lngCodeFormateur

End If

' Désactivation des messages système
DoCmd.SetWarnings False
' Suppression de toutes les données de la table d'impression
DoCmd.RunSQL strSqlDeleteImpressionPlanning

' Instanciation des jeux d'enregistrements
Set rsImprimerPlanning = CurrentDb.OpenRecordset(strSqlImprimerPlanning)

If boolAffichePlanningSalle = True Then
    Set rsTypeImprimer = CurrentDb.OpenRecordset(cstSalles)
Else
    Set rsTypeImprimer = CurrentDb.OpenRecordset("SELECT * FROM T_Formateur " &
strSqlFormateurImprime)
End If

If boolAffichePlanningSalle = True Then
    Set rsPlanning = CurrentDb.OpenRecordset(cstPlanning & strSqlPlanning)
Else
    Set rsPlanning = CurrentDb.OpenRecordset(cstPlanningFormateur & strSqlPlanning)
End If

' Début de la boucle de traitement des salles
Do While Not rsTypeImprimer.EOF
    ' Récupère le CodeSalle dans l'enregistrement rsSalles en cours
    lngCodeSalleEnCours = rsTypeImprimer.Fields(0)
    ' Début de boucle sur la date de planning
    For varDateTraitee = CDate(varDateDebutPlanning) To varDateFinPlanning

```

```

        ' 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 date de résa à ajouter dans la table T_Impression
    ' On saute donc directement au traitement Date WeekEnd ou non (si varDateTraitee = date d'un weekend)
    If boolEOFPlanning = True Then
        ' Branchement à l'étiquette "Reprise"
        ' Ajoute un enregistrement pour Date WeekEnd ou Date sans résa
        GoTo Reprise
    Else
        ' Test pour la concordance entre la salle en cours de traitement et le nom de la salle concernée
        ' par une réservation.
        ' Si différent, branchement à "Reprise" pour dessiner les pavés blancs ou jaunes
        If rsPlanning.Fields(8) <> lngCodeSalleEnCours Then
            GoTo Reprise
            ' Test de concordance entre la date réservée et la date traitée
            ElseIf rsPlanning.Fields(1) = varDateTraitee Then

'Ajoute un enregistrement avec la couleur du produit (Réservation salle)
                rsImprimerPlanning.AddNew
                rsImprimerPlanning.Fields(0) = lngCodeSalleEnCours
                rsImprimerPlanning.Fields(1) = CDate(varDateTraitee)
                rsImprimerPlanning.Fields(2) = rsPlanning.Fields(6)
                rsImprimerPlanning.Update
                ' Test de controle de fin de fichier
                If rsPlanning.EOF Then
                    boolEOFPlanning = True
                Else
                    ' Passe à l'enregistrement suivant
                    rsPlanning.MoveNext
                End If
            End If
        Else
Reprise:
            ' Recherche si la date en cours de traitement est un jour de weekend
            If Weekday(varDateTraitee) = vbSaturday Or Weekday(varDateTraitee) =
vbSunday Then
                'Ajoute un enregistrement avec la couleur weekend
                rsImprimerPlanning.AddNew
                rsImprimerPlanning.Fields(0) = lngCodeSalleEnCours
                rsImprimerPlanning.Fields(1) = CDate(varDateTraitee)
                rsImprimerPlanning.Fields(2) = cstCouleurWeekEnd
                rsImprimerPlanning.Update
            Else
                ' Ce n'est pas un jour de Weekend.
                'Ajoute un enregistrement avec la couleur blanche
                rsImprimerPlanning.AddNew
                rsImprimerPlanning.Fields(0) = lngCodeSalleEnCours
                rsImprimerPlanning.Fields(1) = CDate(varDateTraitee)
                rsImprimerPlanning.Fields(2) = cstCouleurVide
                rsImprimerPlanning.Update
                ' Fin du test sur date traitée
            End If
            ' Fin test sur journée réservée
            End If
        End If
    Next
    'Fin Boucle Salle
    rsTypeImprimer.MoveNext
Loop

    ' Réactivation des messages système
    DoCmd.SetWarnings True

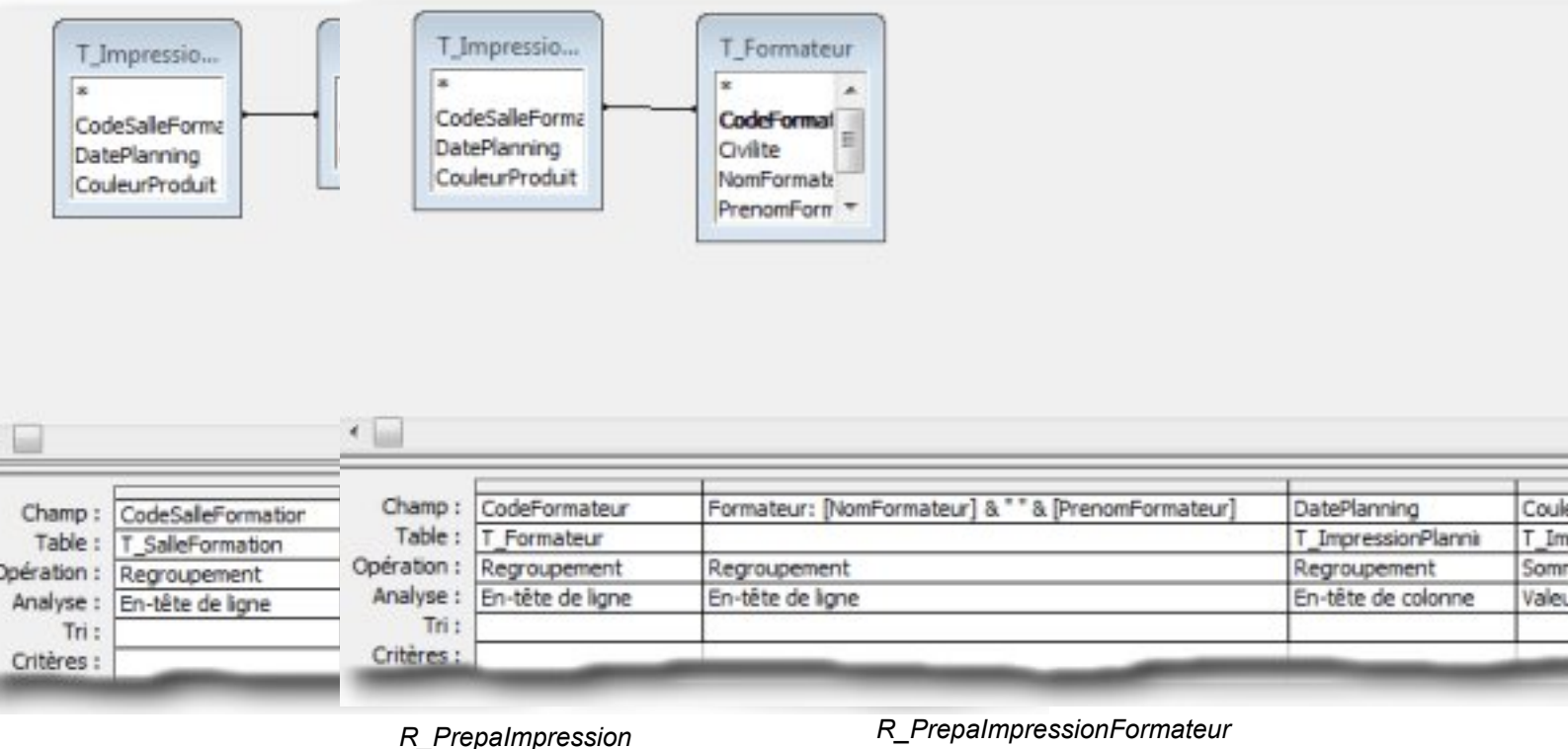
    ' Ouvre le planning en mode aperçu
    DoCmd.OpenReport "E_Planning", acViewPreview
End Sub

```


On remarquera que la dernière instruction de la procédure ouvre l'état en mode aperçu, ce qui entraînera l'activation des diverses procédures de mise en place des sous-états. Nous avons vu précédemment les procédures du sous-état SE_DetailFormation, voyons maintenant celles concernant le sous-état SE_Planning.
Dans un premier temps voici la procédure affectée à l'ouverture du sous-état.

```
Private Sub Report_Open(Cancel As Integer)
    If boolAffichePlanningSalle = True Then
        ' Récupération des données pour alimenter les entêtes de colonnes
        Set rsImpression = CurrentDb.OpenRecordset("R_PrepaImpression")
    Else
        ' Récupération des données pour alimenter les entêtes de colonnes
        Set rsImpression = CurrentDb.OpenRecordset("R_PrepaImpressionFormateurs")
    End If
End Sub
```

Cette procédure a pour objet d'affecter une requête source à notre état. Celle-ci pourra être une des deux requêtes "Analyse croisée" dont voici les structures :



i On remarquera dans la requête *R_PrepaImpressionFormateur*, la concaténation du nom et du prénom du formateur.

Voyons maintenant la procédure qui est affectée à l'évènement "au formatage" de l'entête d'état :

```
Private Sub EntêteEtat_Format(Cancel As Integer, FormatCount As Integer)

    ' initialisation du compteur de champ
    intCompteur = 2

    With rsImpression
        ' Initialisation des étiquettes du planning
        lblTitre1.Caption = "Planning du " & Format(varDateDebutPlanning, "dd mmm yy") & " au " &
        Format(varDateFinPlanning, "dd mmm yy")
    End With
End Sub
```

```

lblTitre2.Caption = "Planning du " & Format(varDateDebutPlanning, "dd mmm yy") & " au " &
Format(varDateFinPlanning, "dd mmm yy")

' Implantation de la date dans les pavés du planning
For Each ctlEtiquette In EnteteEtat.Controls
    If Left(ctlEtiquette.Name, 7) = "lblJour" Then
        ctlEtiquette.Caption = Format(.Fields(intCompteur).Name, "dd mmm yy")
        intCompteur = intCompteur + 1
    End If
Next
End With
rsImpression.MoveFirst
End Sub

```



Deux remarques :

Les étiquettes des dates sont implantées via une boucle sur le nom des champs du jeu d'enregistrements.

On peut voir deux contrôles "lblTitre". En fait, il s'agit là simplement de deux étiquettes légèrement décalées pour donner un effet de relief.

Enfin, abordons la dernière procédure, celle qui est attachée "au formatage" de la section Détail de l'état.

```

Private Sub Détail_Format(Cancel As Integer, FormatCount As Integer)

' Récupération du nom de la salle de formation ou du nom du formateur (puisque'on utilise la même table : T_Impr
txtNomSalle.Caption = rsImpression.Fields(1)
' initialisation du compteur de champ
intCompteur = 2
With rsImpression
    ' Boucle de récupération des codes couleur et affectation à chaque pavé de l'état
    For Each ctlEtiquette In Détail.Controls
        If Left(ctlEtiquette.Name, 10) = "lblIndispo" Then
            ' Gestion d'une éventualité d'une valeur Null - application de la couleur blanche
            ctlEtiquette.BackColor = Nz(.Fields(intCompteur), 16777215)
            ' incrémentation du compteur
            intCompteur = intCompteur + 1
        End If
    Next
    .MoveNext
End With
End Sub

```



Deux remarques : On initialise la variable `intCompteur` à 2 puisque le premier champ contenant un code couleur est le troisième champ du jeu d'enregistrements.

On utilise ici la valeur du champ que l'on affecte à la propriété "BackColor" de chaque pavé du planning.

VI - LA NOTIFICATION

VI-A - Présentation

Nous allons maintenant développer la partie Notification.

Nous allons créer un formulaire permettant de sélectionner le formateur ainsi qu'une période (comprise entre deux dates). A partir de ce formulaire, nous utiliserons la méthode **SendObject**, méthode qui permet d'envoyer un objet par la messagerie , en l'occurrence ce sera un état filtré sur le formateur et la période.


VI-A-1 - Le Formulaire



Voici à quoi ressemblera le Formulaire de notification.


Par défaut les dates et la semaine correspondent à la semaine en cours lors de l'ouverture du formulaire. La liste des formateurs est conditionnée, dans un premier temps par la période et dans un second temps par les formateurs possédant une adresse Mail. Si rien ne s'affiche c'est qu'il n'y a aucune formation pour les formateurs possédant un E-mail.

VI-A-2 - Fonctionnement du formulaire

 Pour envoyer le planning correspondant à un formateur, il vous suffit de sélectionner le formateur et de cliquer sur le bouton Valider. L'état correspondant au planning du formateur de la période sélectionnée est placé en tant que pièce jointe dans un nouvel E-mail. Dans la fenêtre Outlook l'expéditeur, le sujet ainsi que le corps du message seront déjà renseignés il vous sera alors possible de modifier ou d'ajouter d'autres informations.

VI-A-3 - Structure du formulaire

VI-A-3-a - Description des contrôles


 Voici donc la description des différents objets contenus dans le formulaire F_Notification.

Nom du Contrôle	Type de Contrôle	Fonction
btnFermerR	image	L' effet relâché : Visible par défaut, sera la première image
btnFermerS	image	L' effet survolé : Placée sous l'image 1, sera visible lors du passage de la souris sur le "bouton relâché".
btnFermerC	image	L' effet cliqué : Placée sous l'image 2, sera rendu visible lorsque l'on appuie sur le bouton de la souris.
btnValiderR	image	L' effet relâché : Visible par défaut, sera la première image de la pile.
btnValiderS	image	L' effet survolé : Placée sous l'image 1, sera visible lors du passage de la souris sur le "bouton relâché".
btnValiderC	image	L' effet cliqué : Placée sous l'image 2, sera rendu visible lorsque l'on appuie sur le bouton de la souris.
txtCacheFormateur	Champ Texte	Ce champ est non visible , il permet de stocker la valeur du formateur pour


		l'exportation de l'état afin de réaliser l'envoi de l'E-mail.
IstSemaine	Zone de liste déroulante	Stocke les semaines de l'année de 1 à 53.
txtDu	Champ Texte	Stocke la date correspondant au lundi de la semaine en cours.
txtAu	Champ Texte	Stocke la date correspondant au vendredi de la semaine en cours.
IstFormateur	Zone de liste	Permet d'afficher le nom des formateurs ayant une adresse mail.

VI-A-3-b - Les propriétés des contrôles

VI-A-3-b-i - La zone de liste déroulante : IstSemaine

 Nous mettrons comme origine Source : Liste Valeurs.
Nous alimenterons cette liste lors de l'ouverture du Formulaire par VBA

VI-A-3-b-ii - Zones de Texte : txtDu et txtAu

 Nous mettrons comme masque de saisie : 00/00/0000;0;_
Ceci permet de définir la forme de la date.

VI-A-3-b-iii - Zone de liste : IsFormateur

Voici ci-dessous la structure de la requête source de la liste :

```
SELECT T_Formateur.CodeFormateur, [Civillite] & " " & [NomFormateur] & " " & [PrenomFormateur] AS
Formateur
FROM T_Formateur INNER JOIN T_Stages ON T_Formateur.CodeFormateur = T_Stages.CodeFormateur
WHERE
((T_Stages.DateStageDebut) Between [Formulaires]![F_Notification]![txtDu] And [Formulaires]![F_Notification]![txtAu])
AND ((T_Stages.CodeFormateur) Is Not Null)
GROUP BY T_Formateur.CodeFormateur, [Civillite] & " " & [NomFormateur] & " " & [PrenomFormateur],
T_Formateur.Email
HAVING ((T_Formateur.Email) Is Not Null);
```

VI-B - Le Code VBA dans le formulaire

Dans un premier temps nous allons exécuter un code permettant d'initialiser :

- les images servant de boutons.
- la liste pour les semaines ainsi que les dates.

Ce code s'exécutera à l'ouverture du Formulaire

```
Private Sub Form_Open(Cancel As Integer)

    Dim i As Long

    '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"

'Cache de la zone pour les formateurs
Me.txtCacheFormateur.Visible = False

'Boucle pour charger la liste de semaines de 1-53
For i = 0 To 53
    lstSemaine.AddItem i
Next i

'calcul et affichage de la semaine et du lundi correspondant ainsi que le vendredi (date)
lstSemaine = DatePart("ww", Date, vbMonday, vbFirstFourDays)
txtDu = CDate(calculDate(Format(Now, "ww")))
txtAu = txtDu + 4

'regénère la liste Formateur
lstFormateur.Requery

End Sub
```

Dans le code précédent nous initialisons les images qui nous servent de bouton, ensuite nous réalisons une boucle pour alimenter la liste de choix utilisée pour les semaines. En dernier lieu nous récupérons la date du lundi de la semaine en cours. Pour cela nous appellerons une fonction située en fin du module du Formulaire. Nous réalisons en fin de code une régénération de la source de la liste des formateurs, la source est conditionnée par la date de début et de fin, par l'existence de formations entre ces dates et enfin par l'existence d'une adresse mail pour le formateur. Si rien n'est affiché dans la liste c'est qu'aucune formation n'est prévue entre ces dates. Voici le code de la fonction qui calcule la date du lundi de la semaine en cours.

```
Function calculDate(sem As String) As Date
'Fonction permettant de connaître la date du lundi de la semaine en cours

'déclaration des variables
Dim calcultemp As Date
Dim calculjour As Long

'calcul du lundi correspondant à la semaine en cours
calculsem = (sem - (DatePart("ww", Date, vbMonday, vbFirstFourDays))) * 7 + Date
calculjour = 1 - DatePart("w", calculsem, vbMonday, vbFirstFourDays)
calculDate = calculsem + calculjour

End Function
```

Nous allons mettre à jour les zones de texte **txtDu** et **txtAu** sur l'évènement "Après MAJ" de liste déroulante **lstSemaine**. Ici nous appellerons également la fonction **calculDate**

```
Private Sub lstSemaine_AfterUpdate()

'après mise à jour de la liste, mis à jour des champs date Du et Au
txtDu = CDate(calculDate(lstSemaine))
txtAu = txtDu + 4
'regénère la source de la liste
lstFormateur.Requery

End Sub
```

Nous allons également mettre du code après la mise à jour des zones de texte **txtDu** et **txtAu**. Celui-ci permettra de régénérer la source de liste **lstFormateur** en fonction des nouvelles dates.


```
Private Sub txtAu_AfterUpdate()
    'régénère la source de la liste
    lstFormateur.Requery
End Sub

Private Sub txtDu_AfterUpdate()
    'régénère la source de la liste
    lstFormateur.Requery
End Sub
```

Maintenant nous allons mettre le code concernant le bouton Valider. Celui-ci sera placé sur l'évènement "Sur Souris relâchée" de l'image btnValiderS

```
Private Sub btnValiderS_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single)
    'déclaration des variables
    Dim varSelection As Variant
    Dim strMail As String

    On Error GoTo errValider

    ' appel la routine de Bascule des boutons
    Call BasculerBouton("btnValiderC", "btnValiderS")

    'boucle pour lister les sélections de la liste des formateurs
    For Each varSelection In lstFormateur.ItemsSelected
        'récupère l'adresse mail
        strMail = DLookup("[Email]", "T_Formateur", "[CodeFormateur]=" &
        lstFormateur.ItemData(varSelection))
        'charge le code formateur pour la requête
        txtCacheFormateur.Value = lstFormateur.ItemData(varSelection)
        'envoi d'un objet (Etat)
        'pour Access <= 2003 exporter au format snapshot, mettre acFormatSNP

        'pour Access 2007 possibilité d'exporter au format Snapshot ou au format pdf (mettre acFormatPDF)
        DoCmd.SendObject acSendReport, "E_Notification", acFormatPDF, strMail, , , _
        "Votre planning du " & txtDu & " au " & txtAu, _
        "Ci-joint votre planning pour la période du " & txtDu & " au " & txtAu
    Next varSelection
    'sortie de la procédure si tout Ok.
Exit Sub

    'Gestion d'erreur
errValider:

    Select Case Err.Number
        Case "2501" 'erreur commande send annulé
            MsgBox "L'E-mail n'a pas été envoyé," & vbCrLf & _
            "Veuillez recommencer ou contacter l'administrateur de la base", _
            vbCritical, cstDVP
        Case Else 'erreur non connue
            MsgBox "Erreur N° " & Err.Number & vbCrLf & "Description : " & _
            & Err.Description, vbCritical, cstDVP
    End Select
End Sub
```

Expliquons un peu ce code :

- Déclarations des variables.
- Appel de la routine d'erreur si le code suivant soulève une erreur, l'appel permet d'intercepter cette erreur.
- Appel de la routine de gestion d'affichage des boutons pour remettre la visualisation nickel.
- Ensuite nous entrons dans une boucle qui liste toutes les sélections de la zone de liste lstFormateur. Ceci permet de récupérer chaque Formateur sélectionné afin de lui envoyer son planning par E-mail.

Dans un premier temps nous récupérons l'adresse Mail du formateur par la méthode DLookup pour plus d'information sur les fonctions de domaine lire [ici](#).

Ensuite nous chargeons la valeur du code formateur dans la zone de texte cachée, ceci permet à la requête construisant l'état de venir piocher dans la zone pour filtrer.

Puis, nous envoyons via la messagerie l'état par la méthode SendObject. Si aucune erreur n'est levée par Access nous sortons de la procédure, dans le cas contraire nous passons par la routine d'erreur.

Si l'erreur correspond à l'erreur 2501 nous affichons un message personnalisé dans le cas contraire nous affichons le code d'erreur ainsi que la description et nous sortons de la procédure.

Sur le bouton Fermer nous mettrons simplement la fermeture du formulaire :

```
Docmd.close
```

VII - CONCLUSION

Nous disposons maintenant d'un outil beaucoup plus complet. L'idée générale de ce tutoriel étant de donner une piste, un moyen de représenter graphiquement un planning, à chacun maintenant de personnaliser ou de s'inspirer de cet exemple pour se lancer dans sa propre gestion.

VIII - TELECHARGEMENT

Pour vous permettre de matérialiser les concepts décrits ci-dessous vous pouvez télécharger la base exemple. Base **Gestion de Planning Version 2**

IX - REMERCIEMENTS

Mes premiers remerciements iront à **Dolphy35** (Morgan Billy) qui a développé la partie "Notification" de cet article.

Ensuite à **Fring** pour son temps passé à une relecture attentive.

Enfin à toute l'équipe de DVP, toujours présente pour conseiller et soutenir.