C# - FICHE PRATIQUE N° 8

Une petite gestion de personnel
Etats Crystal Report.

A/ Présentation de Crystal Report

Crystal Report est un outil permettant de réaliser des états en interrogeant une source de données quelconque (base de données SQL Server, Access, …).

Le principe est de se connecter à une base de données, de décrire les données à imprimer ainsi que leur présentation. Le résultat de cette étape est un fichier d'extension rpt qui peut ensuite être utilisé pour produire l'état lui-même.

Cet outil qui existe de manière indépendante de tout environnement de développement a été intégré à Visual Studio pour devenir l'outil de production d'états de la plateforme DotNet.

Le processus de réalisation d'un état se déroule en deux temps :

- Définition de l'état à l'aide d'un assistant graphique, cette étape conduit à la production de deux fichiers :

- . Un fichier d'extension rpt qui contient la description de l'état dans un format propriétaire.
- . Un fichier d'extension cs qui contient le code Csharp (très sommaire) de définition d'une classe correspondant à l'état, cette définition s'appuyant sur la description contenue dans le fichier d'extension rpt.

- Production de l'état proprement dit, cet état étant un objet de la classe définie à l'étape précédente. La production de l'état nécessite un composant CrystalReportViewer lié par programme à l'objet état.

B/ Réalisation d'un état

Nous allons réaliser un état présentant service par service les employés dont le salaire est supérieur à 1500 €.

- Ajoutez un nouvel élément de type « état Crystal Report » et nommez-le Rp_employe.
- Sélectionnez l'expert Standard et cliquez sur ok. L'assistant présente une fenêtre dont chaque onglet correspond à l'une des étapes de définition de l'état.
- Sélectionnez votre DataSet dans le dossier « Données du projet » et insérer les deux tables. Remarque, on sélectionne ici le type correspondant au DataSet dbDs_empSce et non l'objet DataSet dbDs_empSce1.
- A l'étape suivante, effectuez la liaison entre les deux tables. Cette liaison peut être faire manuellement à l'aide de la souris ou de manière automatique (liaison par clé).
- Etape suivante : ajoutez les champs numéro, nom, prénom, salaire et désignation du service.
- A l'étape groupe, choisissez de grouper par code et désignation de service.
- L'onglet total permet de réaliser des cumuls. Demandez à compter le nombre de numéros d'employés et à faire la somme de leurs salaires.
- L'étape suivante propose de trier les groupes en fonction des cumuls effectués. Ne faites rien.
- L'étape suivante permet de définir un graphique. Créez un histogramme présentant la comparaison des masses salariales de chaque service.

- L'onglet « sélectionner » permet d'opérer une sélection sur les enregistrements concernés. Ne retenez que les employés gagnant plus de 1 100 €.
- Entrez enfin le titre de l'état et cliquez sur terminer.

L'état est créé, le menu contextuel associé aux différents objets permet d'en modifier assez facilement l'apparence ou le contenu.

C/ Exploitation de l'état

1. Impression de l'état

Créez un nouveau formulaire Fm_report, et placez-y un contrôle « CrystalReportViewer » occupant l'ensemble de la fenêtre (voir la propriété Dock). Nommez-le « rv_report ».

Ce formulaire doit avoir accès au DataSet du formulaire principal. Il faut donc le doter d'un constructeur semblable à celui des formulaires précédents :

```
dbDs_empSce dbDs;
public Fm_report(dbDs_empSce p_dbDs):this()
{
    dbDs=p_dbDs;
}
```

Etant donné que le formulaire ne contient pas de bouton fermer, il faut l'instancier à chaque ouverture. Cela peut être fait à l'aide d'un bouton du formulaire principal :

```
private void bt_report_Click(object sender, System.EventArgs e)
{
    Fm_report fmr;
    fmr=new fm_report(this.dbDs_empSce1);
    fmr.Show();
}
```

Dans le constructeur du formulaire, il faut ajouter le code permettant la création de l'état :

- Déclaration et instanciation d'un objet état de la classe Rp_employe définie à l'étape précédente.

- Association du DataSet dbDs à l'état en tant que source des données.

- Association de l'état au composant CrystalReportViewer.

On obtient au final le constructeur ci-dessous :

```
public Fm_report(dbDs_empSce p_dbDs):this()
{
    dbDs=p_dbDs;
    Rp_employe rp=new Rp_employe();
    rp.SetDataSource(dbDs);
    rv_report.ReportSource=rp;
}
```

Exécutez l'application et imprimez votre état, puis revenez sur sa définition (fichier Rp_employe.rpt) pour améliorer la présentation.

2. Paramétrage de l'état

> Revenez sur la définition de l'état et modifiez la formule de sélection des enregistrements à l'aide du menu contextuel (supprimer cette formule).

Dans le constructeur de Fm_report, renseignez la formule de sélection de manière dynamique :

```
public Fm_report(dbDs_empSce p_dbDs):this()
{
    …
    rp.RecordSelectionFormula="{tp1_employe.salaire} > 1100.00";
    rv_report.ReportSource=rp;
}
```

Remarques :

- Il s'agit d'une chaîne, la formule peut donc aisément être construite par programme (à la suite d'une saisie de l'utilisateur par exemple).
- En ce qui concerne la syntaxe, le mieux est de s'inspirer de l'éditeur de formules…

> Revenez à la définition de l'état et affichez si nécessaire l'explorateur de champs (icône de gauche dans la barre d'outils Crystal Report). Nous allons paramétrer l'état pour fixer plus facilement la limite basse des salaires souhaitée.

Ajoutez un champ de type paramètre nommé « limite » : invite « Entrez la limite : », type nombre, valeur par défaut 1000 (Faites pour cela un clic droit sur « Champs de paramètre » dans l'explorateur de champs).

Modifiez la formule de sélection des enregistrements à l'aide du menu contextuel pour obtenir les employés dont le salaire est supérieur à la limite fixée en paramètre.

Supprimez le titre de l'état et remplacez-le par un objet texte: « salaires supérieur à : ». Accolez-y le champ paramètre (simple drag and drop depuis l'explorateur de champs). Donnez une taille convenable à ces deux éléments et mettez-les en gras.

Supprimez la modification de la formule de sélection dans le constructeur de Fm_report et exécutez l'application : l'utilisateur est invité à saisir la valeur du paramètre.

> Allons un peu plus loin en fixant la valeur de ce paramètre par programme, ce qui fournit plus de souplesse.

Supprimez la formule de sélection établie en mode conception et modifiez le constructeur de Fm_report de la manière suivante après avoir ajouté une directive « using CrystalDecisions.Shared; » à votre code :

```
public Fm_report(dbDs_empSce p_dbDs):this()
{
    dbDs=p_dbDs;
    Rp_employe rp=new Rp_employe();
    rp.SetDataSource(dbDs);
    rv_report.ReportSource=rp;
// créer le paramètre limite
    ParameterFields parametres = new ParameterFields ();
    ParameterField param = new ParameterField ();
    ParameterDiscreteValue valeur = new ParameterDiscreteValue ();
    param.ParameterFieldName = "limite";
    valeur.Value = 1100;
    param.CurrentValues.Add(valeur);
    parametres.Add (param);
    rv_report.ParameterFieldInfo = parametres;
}
```

Le principe est en fait de construire une liste de paramètres (ParameterFields) pour l'état, d'y ajouter le paramètre « limite » (il pourrait y en avoir d'autres), et de fournir cette liste au composant CrystalReportViewer. Les possibilités sont nombreuses, les paramètres pouvant également être de type « fourchette » avec une valeur minimale et une valeur maximale. Consultez la rubrique « paramètres, Crystal Reports » dans l'aide pour en savoir plus…