CONCEPTION D'UN COMPOSANT PERSONNALISE SOUS C# (UNE JAUGE)

Table des matières

I Introduction

A travers ce tutoriel je vais vous montrer comment réaliser un « user control » en C#. Ce composant va permettre d'afficher un graphique représentant une jauge comme celle-ci :

Dans un premier temps nous allons définir les propriété et méthode permettant la gestion de l'affichage graphique des données reçu via le protocole SNMP . Dans un deuxième temps nous allons ajouter au composant un timer qui permettra de redessiner le composant à intervalle régulier.

II Création de la classe Tonerjauge

1. Ajout d'un nouvel élément contrôle utilisateur

Avant de commencer il est nécessaire d'ajouter à votre projet un nouvel élément de type contrôle utilisateur. Vous le nommerez comme bon vous semble. Personnellement je l'ai nommé « tonerjauge.cs ». Nous sommes maintenant prêts à coder notre classe composant utilisateur pour que le résultat soit celui représenté plus haut.

2. Définition de l'interface du contrôle

Pour réaliser ce contrôle vous aurez besoin pour l'instant de ces éléments :

- Un label qui nous permettra d'afficher la légende du graphique, vous pouvez lui donner « lblCaption » comme nom ;
- Un pictureBox qui permettra de dessiner graphiquement notre jauge.

Voici le résultat final dans le concepteur:

Donc maintenant il nous reste plus qu'à coder notre composant.

3. Codage du composant

Nous aurons besoin des propriétés suivante :

- Caption qui nous permettra d'afficher un texte sur le label. Comme par exemple « Toner noir : » ;
- CurrentValue qui va nous permettre de récupérer la valeur en pourcentage du contenu restant. Cette propriété va nous permettre de mettre à jour la jauge graphique ;
- Couleur qui va déterminer la couleur du tracé pour notre jauge.

Voilà ce dont nous avons besoin pour le moment, je vous donne le code correpondant :

```
public partial class tonerjauge : UserControl
    {

        private int currentValue;
        private string caption;
        private string couleur;

        private Printer prn = null;
        private TonerPrinter tp = null;

        public string Caption
        {
            set
            {
                nom = value;
                lblCaption.Text = caption;
            }
            get
            {
                return caption;
            }
        }

        public string Couleur
        {
            set
            {
                couleur = value;
            }

        }


        public int CurrentValue
        {
            set
            {

                currentValue = value;
                //pictureBox1.Invalidate();
            }
            get
```

```
            {
                return GetCurrentValue();
            }
        }
```

Il nous faut de même une référence à l'objet Printer et à l'objet TonerPrinter, car nous avons besoin de récupérer le nom de l'imprimante ainsi que son adresse IP mais aussi l'état des consommables de l'imprimante en question. Voici les déclarations utiles :

```
public TonerPrinter Tp
        {
            set
            {
                tp = value;
            }
        }

        public Printer Prn
        {
            set
            {
                prn = value;
            }
        }
```

Maintenant nous allons définir les méthodes de classe qui vont nous servir à récupérer les valeurs des niveaux des toners et de dessiner notre jauge graphique en conséquence…

Voici le code qui permet de récupérer la valeur de CurrentValue…

```
public int GetCurrentValue()
        {
            int currentlevel = 0;
            TonerPrinter tp = new TonerPrinter();

            if (caption == "Toner noir :")
                currentlevel =
Convert.ToInt32(tp.getTonerStatus(prn.IpAddress, prn.Nom, 1));

            if (caption == "Toner cyan :")
                currentlevel =
Convert.ToInt32(tp.getTonerStatus(prn.IpAddress, prn.Nom, 3));

            if (caption == "Toner magenta :")
                currentlevel =
Convert.ToInt32(tp.getTonerStatus(prn.IpAddress, prn.Nom, 4));

            if (caption == "Toner jaune :")
                currentlevel =
Convert.ToInt32(tp.getTonerStatus(prn.IpAddress, prn.Nom, 2));
```

```
            return currentlevel;
        }
```

Il est à noter que l'on doit récupérer les valeurs pour chaque toner. On se sert de la légende pour récupérer la valeur des OIDs correspondantes…

Venons-en à la méthode permettant de dessiner notre jauge graphique. Voici le code :

```
private void pictureBox1_Paint(object sender, PaintEventArgs e)
        {
            int largeurlevel;
            //this.Invalidate();
            base.OnPaint(e);
            Graphics g = e.Graphics;
            SolidBrush brContour;

            g.Clear(Color.White);
            try
            {
                if (currentValue != -1)
                {
                    pictureBox1.Show();
                    lblCaption.Text = nom;
                    SolidBrush brString = new SolidBrush(Color.Black);
                    SolidBrush br = new SolidBrush(Color.White);
                    g.FillRectangle(br, new Rectangle(0, 0, pictureBox1.Width
- 1, pictureBox1.Height - 1));

                    switch (couleur)
                    {

                        case "Black":
                            brContour = new SolidBrush(Color.Black);
                            break;
                        case "Cyan":
                            brContour = new SolidBrush(Color.Cyan);
                            break;
                        case "Magenta":
                            brContour = new SolidBrush(Color.Magenta);
                            break;
                        case "Jaune":
                            brContour = new SolidBrush(Color.Yellow);
                            break;
                        default:
                            brContour = new SolidBrush(Color.Black);
                            break;
                    }

                    if (currentValue == 0)
                    {
                        currentValue = 100;
                        brContour = new SolidBrush(Color.Red);
                    }
                    Pen pen = new Pen(brContour);
```

```
                    g.DrawRectangle(pen, new Rectangle(0, 0,
pictureBox1.Width - 1, pictureBox1.Height - 1));

                    largeurlevel = (pictureBox1.Width * currentValue) / 100;
                    g.FillRectangle(brContour, new Rectangle(0, 0,
largeurlevel, pictureBox1.Height - 1));
                }
                else
                {
                    pictureBox1.Hide();
                    //g.Clear(Color.White);
                    lblCaption.Text = "Informations indisponibles";
                }

            }
            catch (Exception)
            {

                lblCaption.Text = "Informations indisponibles";
            }
        }
```

Pour ce faire nous éditons la méthode Paint() de l'objet PictureBox ajout au contrôle personnalisé... Pour dessiner dans un PictureBox, il faut que l'on est accès à son contexte graphique. Il faut ensuite définir le style de la brosse qui va nous permettre d'effectuer notre dessin… Il faut définir aussi un objet Pen pour le tracé…

Comme pour la récupération des valeurs du contenu de chaque toner, nous utilisons la valeur de la propriété Caption pour connaître la couleur du tracé… En fait le trac de la jauge n'est possible que si la valeur du contenu restant dans le toner est différente de -1. Dans ce cas précis on affiche dans le label « informations indisponibles ». Un autre cas caractéristique intervient lorsque la valeur est nulle. On dessine alors un rectangle de couleur rouge pour bien montrer que le toner doit être remplacer…

Voilà tout ce dont il est nécessaire de coder pour avoir un composant personnalisé de type jauge… Maintenant si nous voulons que cette jauge puisse être rafraichie à intervalle de temps régulier il est nécessaire de lui ajouter un timer.

4. Ajout d'un timer

Il suffit pour cela d'ajouter un composant timer sur la fiche de composant. Celui-ci apparaîtra dans la partie des composants non visuels de la fenêtre de conception… Le principe est simple lors du lancement de l'application et lors de la création du composant personnalisé le timer est automatiquement démarré par l'intermédiaire de cette méthode:

```
public void StartTimer()
        {
            timer1.Enabled = true;
            timer1.Start();
```

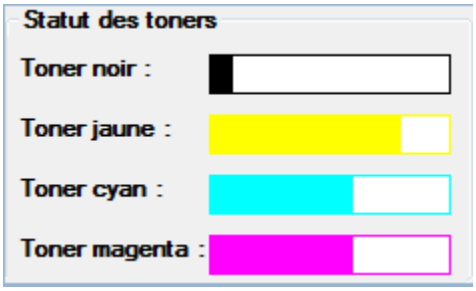
```
        }
```

Lorsque le timer est démarré et à intervalle régulier on demande le rafraîchissement par l'intermédiaire de la méthode suivante :

```
private void timer1_Tick(object sender, EventArgs e)
        {
            currentValue = GetCurrentValue();
            pictureBox1.Refresh();
        }
```

5. Utilisation du contrôle personnalisé

L'utilisation de ce contrôle personnalisé est somme toute assez simple. Elle va nous permettre de réaliser un panel de visualisation de l'état des toners correspondant à chaque imprimante de notre réseau, comme ceci :



Pour la création de chaque jauge nous employons le code suivant :

```
int currentlevel;

//Obtention du niveau courant du toner
currentlevel = Convert.ToInt32(tp.getTonerStatus(adresseIP, nom, 4));

//Création du contrôle jauge Magenta avec initialisation de ces propriétés
tjMagenta.Nom = "Toner magenta :";
tjMagenta.Couleur = "Magenta";
tjMagenta.CurrentValue = currentlevel;

//Placement du contrôle personnalisé sur la groupbox
tjMagenta.Top = 105;
tjMagenta.Left = 1;
//Le contrôle est ajouté à la groupbox
gbxTonersStatus.Controls.Add(tjMagenta);

//On démarre le timer du contrôle...
tjMagenta.StartTimer();

// On transfère les données concernant l'objet imprimante vers la propriété
tjMagenta.Prn = prn;
```

```
// On transfère les données concernant les données du toner vers la propriété correspondante
tjMagenta.Tp = tp;
```

III CONCLUSION

Nous avons vu par l'intermédiaire de ce tutoriel, la conception d'un composant personnalisé. Visual Studio nous facilite grandement la tâche avec son concepteur visuel. Nous avons donc juste à nous préoccupper du codage de notre composant… Cet article est la suite logique de mes deux derniers tutoriels sur la gestion des imprimantes réseaux avec C#. Pour récupérer les données concernant la valeur courante du toner il est nécessaire d'avoir pris connaissance du tutoriel sur la gestion des imprimantes réseau en utilisant OLEPRNLib.dll… Le rafraichissement du contrôle se fait via un timer, par ce procédé la PictureBox se redessine à chaque intervalle, en même tant que le contrôle…