Cours WinDev Numéro 9

Objectifs : Le Champ Webcam.

Utilisation du champ Webcam
Utilisation du potentiomètre.
Les fonctions de dessins.
Les fichiers .ini
Création et utilisation de fichiers image.

Pré-requis : Avoir une Webcam ?

Quel intérêt pédagogique peut-il y avoir à utiliser le champ Webcam ? En fait, ce support va montrer à travers un exemple ludique l'utilisation d'une Webcam pour faire de la détection d'état.
À travers cet exemple, plusieurs thèmes vont être abordés : la création de fichiers image, l'utilisation de fonctions de dessins, l'utilisation du RVB, la création de fichiers de configuration…

Cet applicatif va être utile pour savoir si une zone de couleur définie est présente. Les usages détournés peuvent être :

- La détection de mouvement.
- Vérifier si un bouchon est présent sur une ligne d'embouteillage.
- Etc…

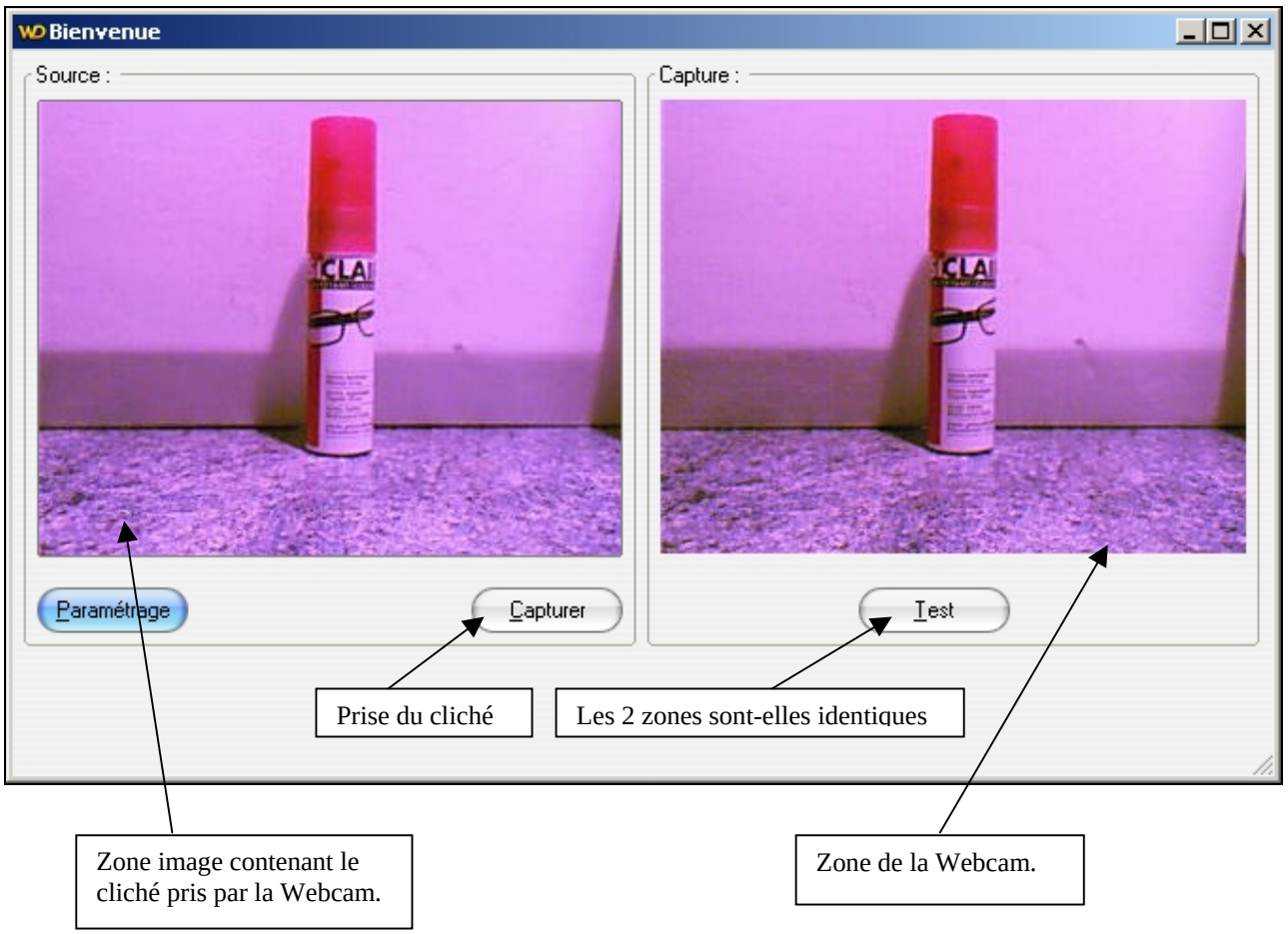
Pour réaliser cet exemple, l'utilisation de la méthodologie objet aurait pu être envisagé, j'ai préféré revenir sur le procédural. Le lecteur motivé pourra regrouper les procédures dans des classes s'il le désire à titre d''approfondissement.

Ce projet se nommera Webcam, et n'aura pas d'analyse. Il y aura 2 fenêtres :

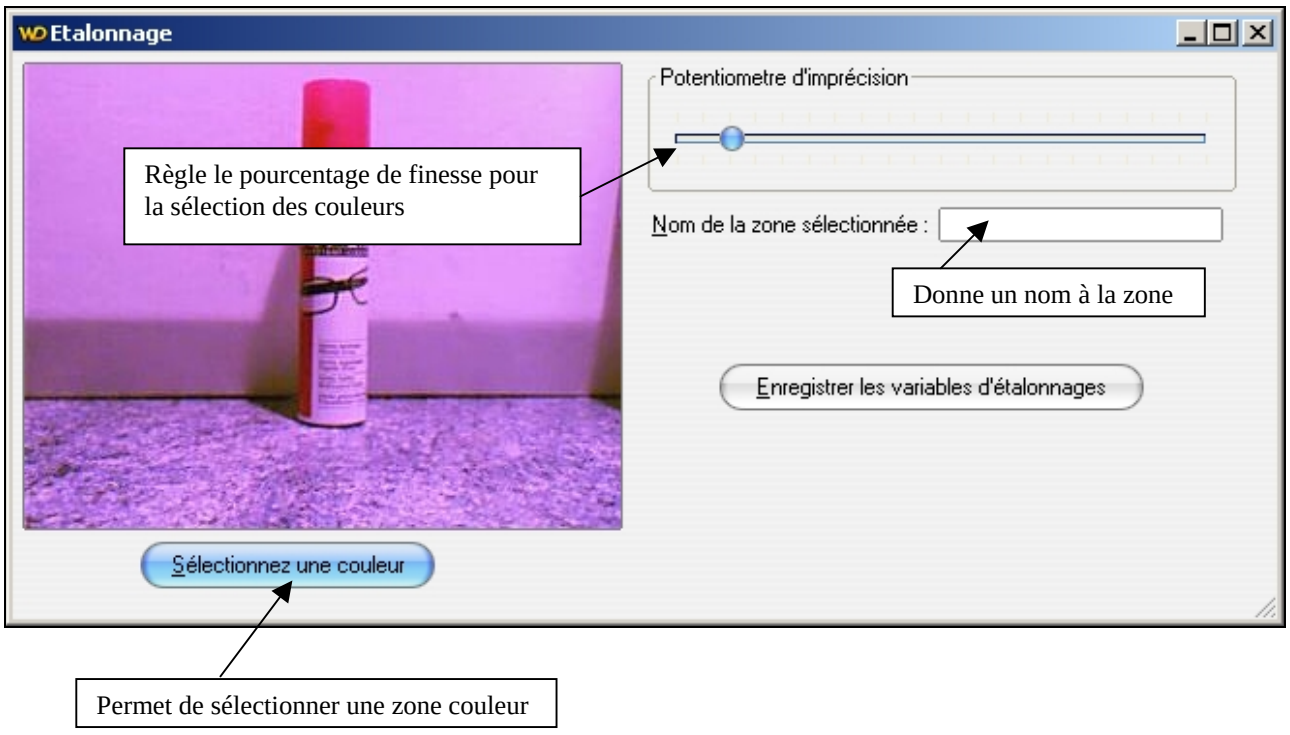
- Départ (la fenêtre initiale de l'application)
- Param (la fenêtre de paramétrage).

Le but de cette application est de faire prendre à la webcam un cliché initial qui servira d'étalon, de sélectionner sur l'étalon la zone dont on va ensuite tester la présence.

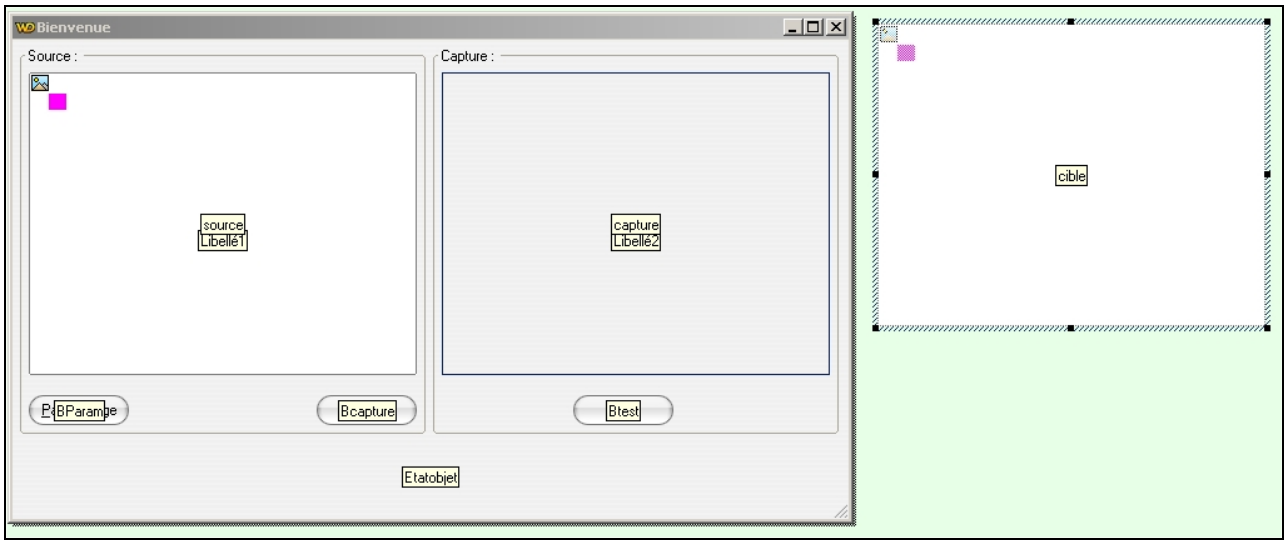
Voici la première fenêtre nommée départ :



Voici le second écran nommé param :



Voici l'écran "Départ" en mode conception :



La fenêtre à comme dimension : 676x410.
Pour connaître ou modifier les dimensions données ici, regardez en bas à droite de l'éditeur WinDev :

On trouve comme info :
Le nom de la fenêtre
La position relative de l'objet dans la fenêtre
La taille de l'objet.
Le plan dans lequel il se trouve.

Pour changer les dimensions vous pouvez modifier directement les valeurs. Commencer par modifier les valeurs de la fenêtre départ

Si vous voulez réécrire la taille le signe de la multiplication est la lettre X et non l'étoile de la multiplication.

Voici les caractéristiques de tous les objets :

Type de champ	Nom	Libelle	Position relative	Taille
Libelle	Par défaut	Source	6,6	324X309
Libelle	Par défaut	Capture	336,6	324X309
Champ de saisie	EtatObjet	Aucun	209,336	250X26
Champ Créer une image	Source	Aucun	12,26	310X242
Champ Web Caméra	Capture	Aucun	343,26	310X242
Bouton	Bparam	Paramètrage	13,284	80X24
Bouton	Bcapture	Capture	243,284	80X24
Bouton	Btest	Test	448,284	80X24
Champ Créer une image	Cible	Aucun	691,-13	310X242

Placez les objets et normalement si vous respectez les consignes votre fenêtre ressemblera à celle représentée en haut de cette page.

Le champ webcam doit être "branché" sur votre Webcam, vous allez le faire manuellement en cliquant avec le bouton droit sur le champ. Choisissez description puis dans l'onglet général faites apparaître votre Webcam dans la combo "Webcam utilisée"

Maintenant, voyons le code de cet écran :

Lorsque l'application se lance s'il y a déjà eu une image "**étalon**" nous allons faire en sorte qu'elle s'affiche dans sa zone image.

Dans la zone de code d'initialisation de la fenêtre départ saisissez ceci :

```
source="etalon.jpeg"
```

Voici le code du bouton **Bcapture** :

```
dSauveImageJPEG(capture,"etalon.jpeg")
source=""
source="etalon.jpeg"
```

La fonction dSauveImageJpeg gele le contenu du champ Webcam "**capture**" et l'enregistre sous "**etalon.jpeg**".
L'instruction suivante enlève le contenu du champ image "**source**"
L'instruction d'après mets l'image "**etalon.jpeg**" dans le champ "**source**"

Voici le code du bouton **Btest** :

```
dSauveImageJPEG(capture,"test.jpeg")
cible=""
cible="test.jpeg"
Compare
```

Vous avez surement vu que je vous ai fait placer une zone image hors fenêtre ? Ce champ image va nous servir au test de comparaison.

Donc, le code du bouton **Btest** fait prendre un cliché appelé "**test.jpeg**", l'affecte au champ image "**cible**" et lance la procédure de comparaison.

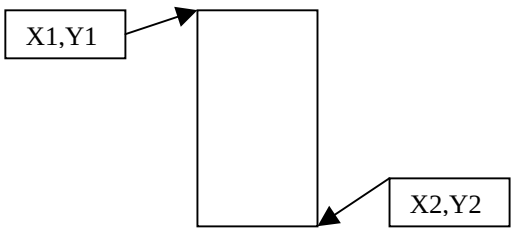
Voyons de suite le contenu de cette procédure compare. Créez une procédure locale nommée compare dont voici le code :

Procédure locale **Compare** Erreur : manuel

```
PROCEDURE Compare()
resultante est un entier
x1,x2,y1,y2 sont des entiers
letableau est un tableau de 6 entiers
trouve est un booléen=Vrai

x1=Val(INILit("Points","X1","",fRepEnCours()+"\ParametresW.ini"))
x2=Val(INILit("Points","X2","",fRepEnCours()+"\ParametresW.ini"))
y1=Val(INILit("Points","Y1","",fRepEnCours()+"\ParametresW.ini"))
y2=Val(INILit("Points","Y2","",fRepEnCours()+"\ParametresW.ini"))

letableau[1]=Val(INILit("Limites","Rouge+","",fRepEnCours()+"\ParametresW.ini"))
letableau[2]=Val(INILit("Limites","Rouge-","",fRepEnCours()+"\ParametresW.ini"))
letableau[3]=Val(INILit("Limites","Vert+","",fRepEnCours()+"\ParametresW.ini"))
letableau[4]=Val(INILit("Limites","Vert-","",fRepEnCours()+"\ParametresW.ini"))
letableau[5]=Val(INILit("Limites","Bleu+","",fRepEnCours()+"\ParametresW.ini"))
letableau[6]=Val(INILit("Limites","Bleu-","",fRepEnCours()+"\ParametresW.ini"))

TANTQUE x1<x2
    resultante=dPixelCouleur(cible,x1,y1)
    trouve=Spectre(resultante,letableau)
    x1+=1
    y1+=1
FIN

SI trouve=Vrai ALORS
    Etatobjet="Le "+INILit("Global","Nom de zone","",fRepEnCours()+"\ParametresW.ini")+" est présent"
SINON
    Etatobjet="Le "+INILit("Global","Nom de zone","",fRepEnCours()+"\ParametresW.ini")+" est absent"
FIN
```

J'en vois qui, rien qu'a la lecture de ce code, claquent des dents, il y en même qui s'évanouissent !

Je vous rappelle le principe du programme.
La fenêtre "**param**" va vous permettre de sélectionner une zone rectangulaire de test, nous aurons donc 4 points (X1,X2,Y1,Y2)



Comme nous ferons des recherches sur les couleurs il faut pouvoir donner un pourcentage (c'est grâce au champ potentiomètre de la fenêtre "param") de spectre rouge acceptable, idem pour le vert et le bleu. C'est pour cette raison que vous verrez des limites bleu-, bleu+ …etc….

Voyons les variables :
Resultante est un entier qui contiendra le code couleur d'un pixel situé à un point (x,y) de la zone image cible et qui servira pour le test de comparaison avec l'image étalon.

X1,X2,Y1,Y2 sont des entiers qui contiennent les points définissants la zone rectangulaire.
Letableau est un tableau qui contient les limites couleur RVB
Trouve est un booléen qui nous indiquera si les couleurs recherchées sont toutes trouvées.

On va placer dans les variables X1,X2,Y1,Y2 les points définissants notre zone rectangulaire de test. Ces points sont dans un fichier Ini nommé ParametresW.ini.
Pour avoir une idée du fonctionnement de INITLit ouvrez l'aide.
Le tableau va être rempli avec les valeurs couleur limites.

```
TANTQUE x1<x2
        resultante=dPixelCouleur(cible,x1,y1)
        trouve=Spectre(resultante,letableau)
        x1+=1
        y1+=1
FIN
```

Les plus rapides ont-ils compris ce que fait ce code ? Pour les autres voici une explication. En fait, on va tracer une diagonale dans notre zone rectangulaire, point par point. Pour chaque point on relève sa couleur dans la variable **résultante**. Cette variable résultante est passée dans la procédure globale **Spectre** qui contrôle si la couleur est comprise dans les limites RVB de références.

Selon le résultat du booléen **trouve**, on affiche un message dans la zone EtatObjet.

Voici le code de la procédure globale "**Spectre**" :

Procédure globale **Spectre**

```
PROCEDURE Spectre(Lacouleur,letableau)
srouge,svert,sbleu sont des entiers
trouve est un booléen=Faux

srouge=modulo(Lacouleur,256)
svert=modulo((Lacouleur/256),256)
sbleu=modulo(((Lacouleur/256)/256),256)

SI letableau[2]<=srouge<=letableau[1] ALORS
    trouve=Vrai
FIN

SI letableau[4]<=svert<=letableau[3] ALORS
    trouve=Vrai
FIN

SI letableau[6]<=sbleu<=letableau[5] ALORS
    trouve=Vrai
FIN

RENVOYER trouve
```

Explication du code :

Pour trouver la valeur RGB comprise dans l'entier renvoyé par la fonction DpixelCouleur on est obligé d'effectuer certains calculs.
Ainsi pour avoir la dominante Rouge on effectue le calcul suivant l'entier couleur modulo 256.

Dans le tableau nous avons nos limites par couleur. Le but est de voir si la couleur récupéree est comprise entre ces 2 valeurs du tableau.

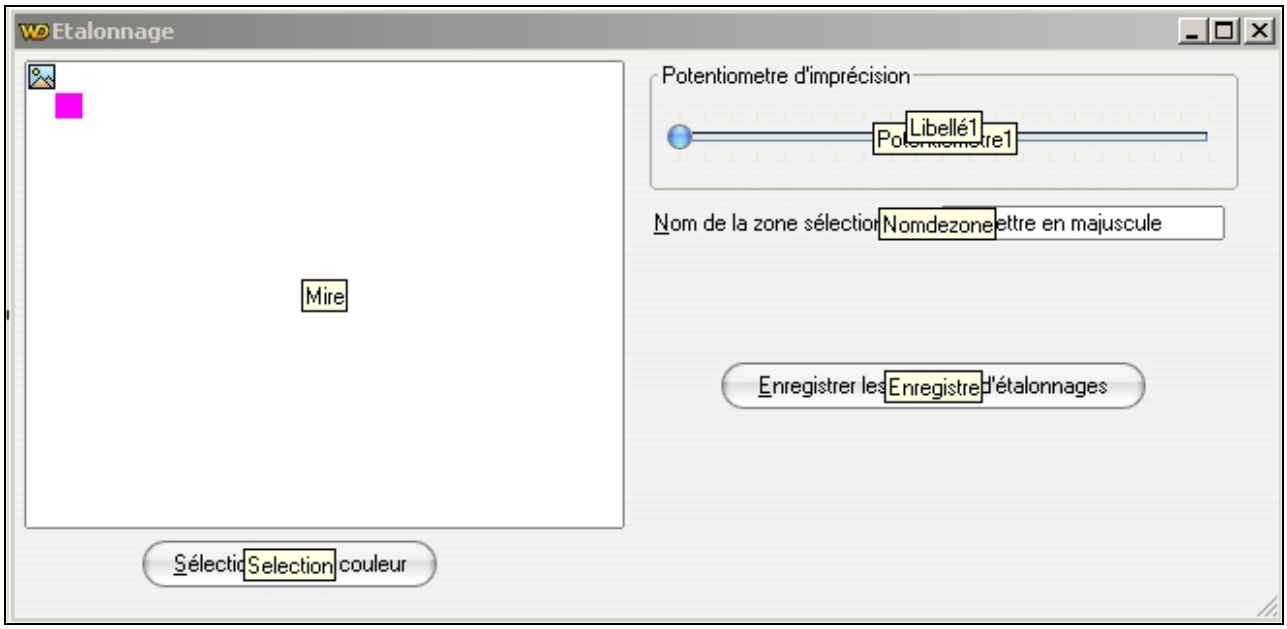
Nous avons un booléen qui selon son état indiquera si la couleur est dans les limites.

Vous aurez une vue d'ensemble lorsque la fenêtre **param** aura été vue.

Voyons le code du bouton **Bparam** :

```
Ouvre(Param)
```

Maintenant que tout le code de la fenêtre départ est décrit passons à la fenêtre Param.



La fenêtre a les dimensions suivantes : 660X320

Voici le tableau de positionnement :

Type de champ	Nom	Libelle	Position relative	Taille
Champ image	Mire	Source	5,5	310X242
Bouton	Selection	Capture	336,6	324X309
Bouton	Enregistre	Enregistrer les variables d'étalonnages	365,161	219X24
Potentiometre	Par défaut		335,26	290X38
Libelle	Par défaut	Potentiometre d'imprécision	328,6	304X66
Champ de saisie	Nomdezone	Nom de la zone sélectionnée :	328,78	297X22

Vous devez avoir maintenant une fenêtre ressemblante à celle présentée ci-dessus.

Dans la zone de déclaration globale de la fenêtre inscrivez ceci :

```
rescouleur,Crouge,Cvert,Cbleu sont des entiers
Lex,ley sont des entiers
letab est un tableau de 6 entiers
```

Le potentiomètre va nous servir pour avoir un pourcentage dans la couleur sélectionnée. Par exemple, si l'utilisateur a cliqué dans un rouge de code RVB 200 et que la valeur retournée par le potentiomètre soit 10, on appliquera 10 % de plus à 200 pour la couleur la plus haute et –10% pour avoir l'intervalle bas.
Ces 2 valeurs seront ensuite enregistrées dans un tableau.

Voici le descriptif du potentiomètre :
Faites un clic droit dessus et choisissez "**Description**" ensuite l'onglet "**Détail** ". Dans la zone **valeur initiale** mettez 10, dans la **valeur minimale** mettez **0** et **100** pour la **maximale**.

Voici le code du bouton sélection :

```
Mire..CurseurSouris = curCroix
```

Comme vous le voyez, ce code ne fait pas grand-chose, il fait en sorte que le curseur de survol de l'image soit une croix.

Voyons maintenant comment s'opère la sélection d'une zone couleur. Pour pouvoir capturer une zone, il faut que l'utilisateur clique dans l'image. Le code sera donc dans la partie clic de la zone image. Le voici :

Clic sur **Mire** *

```
MoiMême="etalon.jpeg"

Lex=SourisPosX
ley= SourisPosY

rescouleur = dPixelCouleur(Mire, Lex, ley)
Crouge=modulo(rescouleur,256)
Cvert=modulo((rescouleur/256),256)
Cbleu=modulo(((rescouleur/256)/256),256)


letab=Spectre_origine(rescouleur,Potentiomètrel..Valeur)

calculzone(Lex,ley,letab)
```

Il faut donc qu'à chaque clic le système nous délimite une zone couleur sur l'image étalon, pour éviter de superposer les rectangles de sélection à chaque clic l'image de base est réécrite dans la zone image.
C'est le rôle de la première ligne.

Ensuite nous récupérons la position exacte de la souris lors du clic.
Puis nous récupérons la valeur couleur du pixel situé à la position de la souris.
Nous calculons les Valeurs RVB.
Il nous faut ensuite remplir le tableau des valeurs couleur du point cliqué. C'est la fonction Spectre_origine qui s'en charge.

Voici son code :

Procédure locale **Spectre_origine**

```
PROCEDURE Spectre_origine(lacouleurOriginelle,coef)
  delta est un entier
  srouge,svert,sbleu sont des entiers
  srougem,svertm,sbleum sont des entiers
  resultat est un tableau de 6 entiers

  srouge=modulo(lacouleurOriginelle,256)
  srougem=srouge
  svert=modulo((lacouleurOriginelle/256),256)
  svertm=svert
  sbleu=modulo(((lacouleurOriginelle/256)/256),256)
  sbleum=sbleu

  delta=(coef/100)*srouge
  srouge+=delta
  srougem-=delta
  resultat[1]=srouge
  resultat[2]=srougem
  delta=(coef/100)*svert
  svert+=delta
  svertm-=delta
  resultat[3]=svert
  resultat[4]=svertm
  delta=(coef/100 )*sbleu
  sbleu+=delta
  sbleum-=delta
  resultat[5]=sbleu
  resultat[6]=sbleum

  RENVOYER resultat
```

Ce code ne sert qu'à remplir un tableau de valeurs correspondantes à un entier sur lequel on applique un pourcentage donné par le potentiomètre.

Voyons maintenant la fonction CalculZone.

```
PROCEDURE calculzone(posx,posy,letableau)
Pxd,pxg,pyb,pyh sont des entiers
trouve est un booléen=Vrai

Pxd=posx
pxg=posx
pyb=posy
pyh=posy

dDébutDessin(Mire)

// on recherche une zone rectangulaire comportant cette couleur
// on commence par les x vers la droite

TANTQUE trouve
	SI Spectre(dPixelCouleur(Mire, Pxd, posy),letableau) ALORS
		Pxd=Pxd+1
	SINON
		trouve=Faux
	FIN
FIN

// on continue les x vers la gauche
trouve=Vrai
TANTQUE trouve
	SI Spectre(dPixelCouleur(Mire, pxg, posy),letableau) ALORS
		pxg=pxg-1
	SINON
		trouve=Faux
	FIN
FIN

// on cherche y vers le bas
trouve=Vrai
TANTQUE trouve
	SI Spectre(dPixelCouleur(Mire, posx, pyb),letableau)ALORS
		pyb=pyb+1
	SINON
		trouve=Faux
	FIN
FIN

// on cherche y vers le haut
trouve=Vrai

TANTQUE trouve
	SI Spectre(dPixelCouleur(Mire, posx, pyh),letableau) ALORS
		pyh=pyh-1
	SINON
		trouve=Faux
```

```
      FIN
FIN

INIEcrit("Points","X1",NumériqueVersChaîne(pxg),fRepEnCours()+"\ParametresW.ini")
INIEcrit("Points","Y1",NumériqueVersChaîne(pyh),fRepEnCours()+"\ParametresW.ini")
INIEcrit("Points","X2",NumériqueVersChaîne(Pxd),fRepEnCours()+"\ParametresW.ini")
INIEcrit("Points","Y2",NumériqueVersChaîne(pyb),fRepEnCours()+"\ParametresW.ini")

dRectangle(pxg,pyh,Pxd,pyb,iTransparent,iNoir)
```

Cette fonction a un rôle important. Lorsque l'on clique sur l'image elle va essayer de trouver une zone rectangulaire comprise dans un spectre, plus ou moins le pourcentage, défini par la couleur de référence du clic. Une fois la zone couleur définie, les valeurs des points ayant servis à faire la zone rectangulaire sont mémorisées dans le fichier ini.

Maintenant voici le code du bouton enregistre :

Clic sur **Enregistre** — Erreur : manuel — Exception : manuel

```
SI Nomdezone<>"" ALORS
    INIEcrit("Global","Nom de zone",Nomdezone,fRepEnCours()+"\ParametresW.ini")
    INIEcrit("Global","Pourcentage",NumériqueVersChaîne(Potentiomètrel..Valeur),fRepEnCours()+"\ParametresW.ini")

    INIEcrit("Points","X",NumériqueVersChaîne(Lex),fRepEnCours()+"\ParametresW.ini")
    INIEcrit("Points","Y",NumériqueVersChaîne(ley),fRepEnCours()+"\ParametresW.ini")

    INIEcrit("Couleurs","Originelle",NumériqueVersChaîne(rescouleur),fRepEnCours()+"\ParametresW.ini")
    INIEcrit("Couleurs","Rouge",NumériqueVersChaîne(Crouge),fRepEnCours()+"\ParametresW.ini")
    INIEcrit("Couleurs","Vert",NumériqueVersChaîne(Cvert),fRepEnCours()+"\ParametresW.ini")
    INIEcrit("Couleurs","Bleu",NumériqueVersChaîne(Cbleu),fRepEnCours()+"\ParametresW.ini")

    INIEcrit("Limites","Rouge+",NumériqueVersChaîne(letab[1]),fRepEnCours()+"\ParametresW.ini")
    INIEcrit("Limites","Rouge-",NumériqueVersChaîne(letab[2]),fRepEnCours()+"\ParametresW.ini")
    INIEcrit("Limites","Vert+",NumériqueVersChaîne(letab[3]),fRepEnCours()+"\ParametresW.ini")
    INIEcrit("Limites","Vert-",NumériqueVersChaîne(letab[4]),fRepEnCours()+"\ParametresW.ini")
    INIEcrit("Limites","Bleu+",NumériqueVersChaîne(letab[5]),fRepEnCours()+"\ParametresW.ini")
    INIEcrit("Limites","Bleu-",NumériqueVersChaîne(letab[6]),fRepEnCours()+"\ParametresW.ini")

    Info("Enregistrement effectué")
    Ferme
SINON
    Info("Veuillez entrer un nom de zone")
FIN
```

Ce code ne fait qu'enregistrer les variables dans un fichier ini qui sera utilisé dans le code de la fenêtre départ pour interpréter l'image.

Voilà le code est fini de saisir, vous pouvez essayer votre application.