Premiers pas avec GNOGA

L'interface utilisateur au sens plus large que interface graphique est le mécanisme d'interaction entre l'utilisateur d'un programme informatique et l'exécutions des instructions de ce programme. Le choix d'une bibliothèque graphique est souvent liée à un langage de programmation et surtout à une plateforme d'exécution : .NET avec C# pour Windows, Swing avec Java pour JVM, Cocoa avec Objective-C voire Swift pour macOS et iOS, GTK avec C pour Linux, etc. Rares sont celles qui ont été conçues pour plusieurs langages incluant notamment Ada. Certains font de la résistance en développant des bibliothèques relais (bindings en anglais) comme QtAda ou GTKAda. La question est alors de pouvoir s'adapter à plusieurs plateformes en gardant une cohérence globale. Java, après avoir essayé avec AWT de conserver les spécificités de chaque plateforme, a préféré n'avoir qu'une seule cohérence la sienne avec Swing. Et Ada dans tout ça pourrait faire ce choix car par essence il indépendant de toutes plateformes d'exécution.

Le standard Ada 2012 est maintenant adopté, il comporte de nombreuses nouveautés. J'espère que celles-ci aiderons à favoriser le choix d'Ada dans les développements informatiques.

L'atout initial d'Ada est l'assurance qu'un code source aura un comportement identique quelque soit la plate-forme d'exécution de part le compilateur (s'il accepte la compilation) mais aussi de part la bibliothèque du standard Ada.

Un projet de standardisation des spécifications d'un ensemble de paquetages Ada pour un environnement graphique multi-fenêtres même minimal, indépendant de la plate-forme d'exécution, respectant ainsi la compatibilité source du langage Ada, serait le bienvenu après la récente standardisation d'Ada 2012. Les implémentations de ces spécifications seraient elles dépendantes des plates-formes.

L'utilité d'une standardisation n'est pas tant d'avoir un choix de plus parmi d'autres avec chacun des avantages et des inconvénients ou que l'on aime ou pas mais bien d'avoir l'assurance d'être utilisable par tous.

Malgré le positionnement du langage Ada principalement dans le monde professionnel peuplé d'ingénieurs où le "fun" (un des paradoxes soulevé par Jean-Pierre Rosen dans "The Ada paradox(es)") n'est pas le critère principal, je pense au contraire que l'apport "natif" d'une bibliothèque graphique apporterait un peu de "fun" qui serait profitable à Ada pour son aura après des jeunes ingénieurs (ou moins jeunes), quitte à pencher vers plus d'instantanéité contraire au méthode de développement logiciel.

Le sujet me semble bien vaste. Mais je sais que cela ne pourra être que bénéfique. À l'inverse, il me semble que son absence est un frein dans l'environnement actuel où tout est interface graphique : la moindre petite démo est mieux valorisée dans une interface graphique habituelle.

Quelques surprises pourraient venir d'applications graphiques orientées Web indépendantes de la plateforme par nature comme le propose la toute nouvelle bibliothèque GNOGA.

Sommaire

1. Introduction

GNOGA est une bibliothèque graphique créée nativement en Ada. Ce n'est pas une sur-couche Ada à une bibliothèque existante en C ou C++. Sa particularité est de permettre de construire des applications graphiques orientées Web indépendantes de la plateforme. Double indépendance garantie d'une part de fait du langage Ada lui même, Ada assure qu'un code source aura un comportement identique quelque soit la plate-forme d'exécution de part son compilateur (s'il accepte la compilation), d'autre part avec l'utilisation du HTML et Javascript pour le rendu graphique dans un navigateur Web.

Site : http://www.gnoga.com.

GNOGA est tout jeune, sa documentation n'est actuellement pas très étoffée. Une FAQ et un README sont présent à la racine de la bibliothèque et quelques textes sur le développement et le déploiement d'applications sont présent dans le répertoire "docs".

Des démonstrations dont le déjà mythique "snake" sont présentes dans le répertoire "demo" et des tutoriels dans le répertoire "tutorial".

2. Installation à partir des sources

GNOGA est actuellement dans une phase de développement non stable mais la version courante est utilisable pour faire des essais. La bibliothèque est disponible sur https://sourceforge.net/projects/gnoga via GIT.

La version stable GNOGA V1.2a est aussi disponible prête à l'emploi dans la bibliothèque XNAdaLib-2016, voir son installation depuis le site de Source Forge "https://sourceforge.net/projects/gnuada/files/GNAT_GPL%20Mac%20OS%20X/2016-el-capitan".

Pour construire la version courante à partir des sources, suivre les instructions suivantes.

Avant de démarrer la compilation, GNAT doit être activé :

```
$ PATH=/usr/local/gnat/bin:$PATH
```

Je conseil de créer un répertoire dédié à la construction des bibliothèques :

```
$ instxada=/usr/local/xnadalib-2016
```

```
$ mkdir -p $instxada
$ cd $instxada
```

Saisir les commandes suivantes dans le Terminal :

```
$ git clone git://git.code.sf.net/p/gnoga/code gnoga-code
$ cd gnoga-code
$ make all
$ make html-docs rm-docs
$ make PREFIX=$instxada install_debug
```

Pour une utilisation temporaire, utiliser à chaque fois les commandes suivantes :

```
$ PATH=$instxada/bin:$PATH
$ export GPR_PROJECT_PATH=$instxada/share/gpr:$GPR_PROJECT_PATH
```

Une documentation sous forme de manuels utilisateur et d'un manuel de référence est disponible dans le répertoire "$instxada/share/gnoga/html".

```
$ open $instxada/share/gnoga/html/user_guide.html
$ open $instxada/share/gnoga/html/api_summary.html
$ open $instxada/share/gnoga/html/gnoga_rm/index.html
```

3. Utilisation basique

Nous allons créer un petit programme qui affiche "Hello World!". Pour cela nous allons invoquer le modèle "hello_world", construire le programme et l'exécuter :

```
$ cd <répertoire de test>
$ gnoga_make new hello hello_world
# les répertoires css, html, img, js et src ont été créés
$ cd hello
$ make
# les répertoires obj et bin ont été créés
$ bin/hello
# une alerte peut apparaitre avec le message:
# "Voulez-vous que l'application « hello » accepte les connexions réseau entrantes ?"
# cliquer sans crainte sur le bouton "Refuser"
# Une fenêtre de navigation Web s'est ouverte avec le message "Hello World!"
# faites <crtl>-C pour arrêter le programme.
```

Gnoga a créé un serveur Web qui interagit avec le programme écrit en Ada via du code JavaScript exécuté par le navigateur Web.

Un projet GPR est présent dans le répertoire "src" pour pourvoir utiliser GPS. Faites comme habituellement Build et Run, et voilà.

4. Construction d'une interface graphique pour calculer une factorielle

a) Ajout du bouton "Quitter"

Nous allons tout d'abord ajouter un bouton "Quitter" pour éviter d'interrompre brutalement l'exécution du programme. La procédure "On_Quit" est connectée au bouton "Quitter" et sera activée à chaque clic.

Éditer le fichier src/hello.adb, ajouter le code en gras :

```
with Ada.Exceptions;

with Gnoga.Application.Singleton;
with Gnoga.Gui.Window;
with Gnoga.Gui.View;
with Gnoga.Gui.Base;
with Gnoga.Gui.Element.Common;

procedure hello is
  Main_Window : Gnoga.Gui.Window.Window_Type;
  Main_View   : Gnoga.Gui.View.View_Type;
  Quit_Button : Gnoga.Gui.Element.Common.Button_Type;

  procedure On_Quit (Object : in out Gnoga.Gui.Base.Base_Type'Class) is
    pragma Unreferenced (Object);
  begin
    Gnoga.Application.Singleton.End_Application;
  end On_Quit;

begin
  Gnoga.Application.Title ("hello");
  Gnoga.Application.HTML_On_Close
   ("<b>Connection to Application has been terminated</b>");
  Gnoga.Application.Open_URL ("http://127.0.0.1:8080");
  Gnoga.Application.Singleton.Initialize (Main_Window, Port => 8080);

  Main_View.Create (Main_Window);
  Quit_Button.Create (Main_View, "Quitter");
  Quit_Button.On_Click_Handler (On_Quit'Unrestricted_Access);
  Main_View.Put_Line ("Hello World!");

  Gnoga.Application.Singleton.Message_Loop;
exception
  when E : others =>
    Gnoga.Log (Ada.Exceptions.Exception_Name (E) & " - " &
               Ada.Exceptions.Exception_Message (E));
end hello;
```

Compiler et exécuter le programme, cliquer sur "Quitter" dans la fenêtre Web qui s'affiche, le programme s'arrête.

Nous allons ajouter d'autres éléments pour construire une calculatrice de factorielles.

b) Ajout d'une zone de saisie

Nous allons ajouter un formulaire avec une zone de saisie d'un nombre entier.
Éditer le fichier src/hello.adb, ajouter le code en gras :

```
<...>
with Gnoga.Gui.Element.Common;
with Gnoga.Gui.Element.Form;

procedure hello is
  Main_Window    : Gnoga.Gui.Window.Window_Type;
  Main_View      : Gnoga.Gui.View.View_Type;
  Quit_Button    : Gnoga.Gui.Element.Common.Button_Type;
  Fact_Form      : Gnoga.Gui.Element.Form.Form_Type;
  Input_Text     : Gnoga.Gui.Element.Form.Text_Type;
  Question_Label : Gnoga.Gui.Element.Form.Label_Type;
<...>
  Quit_Button.On_Click_Handler (On_Quit'Unrestricted_Access);
  Fact_Form.Create (Main_View);
  Input_Text.Create (Fact_Form, 40);
  Question_Label.Create (Fact_Form, Input_Text, "Saisir un nombre entier : ");
<...>
end hello;
```

Compiler et exécuter le programme, une zone de saisie s'affiche avec son texte d'invite.

c) Ajout d'un bouton de calcul et d'une zone de résultat

Nous allons ajouter un bouton "Calcul" connecté à la procedure "On_Factorial" avec une zone texte pour le résultat.
Éditer le fichier src/hello.adb, ajouter le code en gras :

```
<...>
  Question_Label : Gnoga.Gui.Element.Form.Label_Type;
  Fact_Button    : Gnoga.Gui.Element.Common.Button_Type;
  Result_Label   : Gnoga.Gui.Element.Common.DIV_Type;

  procedure On_Quit (Object : in out Gnoga.Gui.Base.Base_Type'Class) is
    pragma Unreferenced (Object);
  begin
    Gnoga.Application.Singleton.End_Application;
  end On_Quit;

  procedure On_Factorial (Object : in out Gnoga.Gui.Base.Base_Type'Class) is
    pragma Unreferenced (Object);
    function Factorielle (N : Natural) return Long_Long_Integer is
      F : Long_Long_Integer := 1;
    begin
      for I in 2 .. N loop
        F := F * Long_Long_Integer (I);
      end loop;
      return F;
    end Factorielle;
  begin
```

```
      Result_Label.Text
      ("Résultat " &
       Input_Text.Value &
       "! = " &
       Long_Long_Integer'Image
         (Factorielle (Natural'Value (Input_Text.Value))));
    exception
      when others =>
        Result_Label.Text ("Erreur !");
    end On_Factorial;
  <...>
    Question_Label.Create (Fact_Form, Input_Text, "Saisir un nombre entier : ");
    Main_View.Put_Line ("Calcul de factorielle :");
    Fact_Button.Create (Main_View, "Calcul");
    Fact_Button.On_Click_Handler (On_Factorial'Unrestricted_Access);
    Result_Label.Create (Main_View);

    Gnoga.Application.Singleton.Message_Loop;
  <...>
  end hello;
```

Ajouter l'option "-gnato" dans le projet "src/hello.gpr", compiler et exécuter le programme puis saisir un nombre et cliquer sur le bouton "Calcul", et voilà le travail :

d) Ajout d'une case à cocher et d'un curseur

Nous allons ajouter à notre formulaire une case à cocher pour choisir un calcul de factorielle en nombres flottants imprécis mais permettant de plus grandes valeurs, notre factorielle entière étant limitée à 20 ! Un curseur permet de choisir le nombre de décimales à afficher.

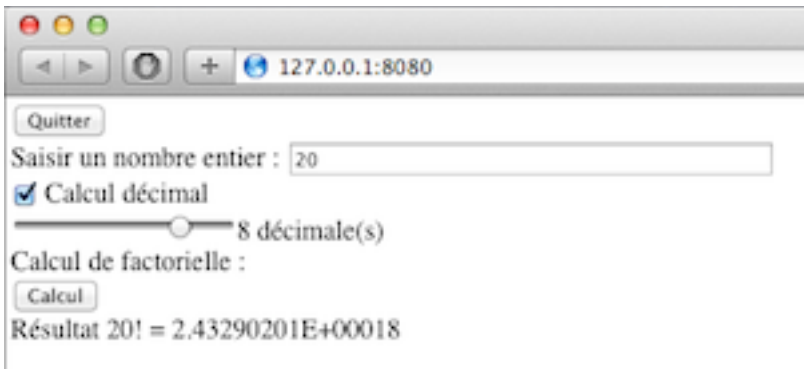
Éditer le fichier src/hello.adb, ajouter le code en gras :

```
  with Ada.Exceptions;
  with Ada.Long_Long_Float_Text_IO;
  <...>
    Result_Label    : Gnoga.Gui.Element.Common.DIV_Type;
    Dec_Check_Box   : Gnoga.Gui.Element.Form.Check_Box_Type;
    Dec_Label       : Gnoga.Gui.Element.Form.Label_Type;
    Dec_Range       : Gnoga.Gui.Element.Form.Range_Type;
    Dec_Range_Label : Gnoga.Gui.Element.Form.Label_Type;
    Dec_Value_Label : Gnoga.Gui.Element.Form.Label_Type;
  <...>
      function Factorielle (N : Natural) return Long_Long_Integer is
        F : Long_Long_Integer := 1;
      begin
        for I in 2 .. N loop
```

```
        F := F * Long_Long_Integer (I);
      end loop;
      return F;
    end Factorielle;
    function Factorielle (N : Natural) return String is
      F : Long_Long_Float := 1.0;
      S : String (1 .. 20);
    begin
      for I in 2 .. N loop
        F := F * Long_Long_Float (I);
      end loop;
      Ada.Long_Long_Float_Text_IO.Put (S, F, Dec_Range.Value, 6);
      return S;
    end Factorielle;
  begin
    if not Dec_Check_Box.Checked then
      Result_Label.Text
      ("Résultat " &
       Input_Text.Value &
       "! = " &
       Long_Long_Integer'Image
        (Factorielle (Natural'Value (Input_Text.Value))));
    else
      Result_Label.Text
      ("Résultat " &
       Input_Text.Value &
       "! = " &
       Factorielle (Natural'Value (Input_Text.Value)));
    end if;
  exception
    when others =>
      Result_Label.Text ("Erreur !");
  end On_Factorial;

  procedure Dec_Range_Change
   (Element : in out Gnoga.Gui.Base.Base_Type'Class)
  is
    pragma Unreferenced (Element);
  begin
    Dec_Value_Label.Text (Dec_Range.Value);
  end Dec_Range_Change;

begin
<...>
  Question_Label.Create (Fact_Form, Input_Text, "Saisir un nombre entier : ");
  Fact_Form.New_Line;
  Dec_Check_Box.Create (Fact_Form);
  Dec_Label.Create (Fact_Form, Dec_Check_Box, " Calcul décimal", False);
  Fact_Form.New_Line;
  Dec_Range.Create (Fact_Form);
  Dec_Range.Minimum (0);
  Dec_Range.Maximum (10);
  Dec_Range.Value (0);
  Dec_Value_Label.Create (Fact_Form, Dec_Range, "0", False);
  Dec_Range_Label.Create (Fact_Form, Dec_Range, " décimale(s)", False);
  Dec_Range.On_Change_Handler (Dec_Range_Change'Unrestricted_Access);
```

```
   Main_View.Put_Line ("Calcul de factorielle :");
<...>
end hello;
```

Voilà le résultat :



e) Contraintes et visibilité

Nous allons ajouter une contrainte sur la saisie pour que l'utilisateur soit averti lors d'une erreur de frappe et désactiver le curseur lorsque qu'il ne sert pas. Nous en profitons pour donner le focus par défaut à la zone de saisie et prendre en compte la touche entrée.

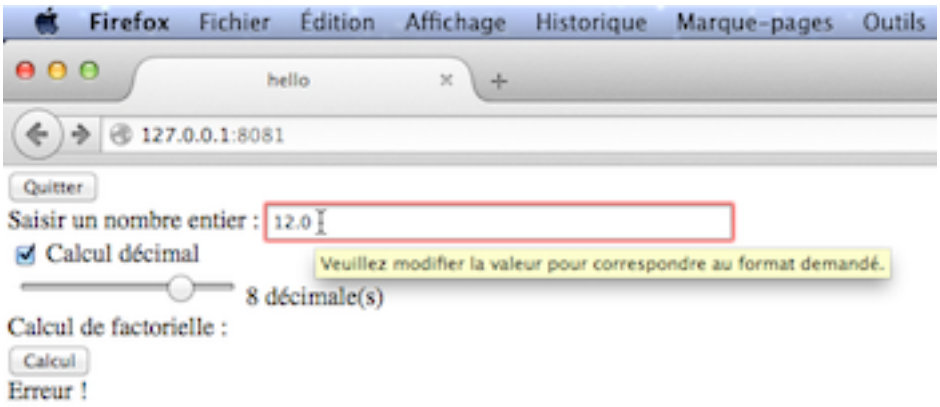
Éditer le fichier src/hello.adb, ajouter le code en gras :

```
<...>
   end Dec_Range_Change;

   procedure On_Dec_Change (Element : in out Gnoga.Gui.Base.Base_Type'Class) is
      pragma Unreferenced (Element);
   begin
      Dec_Range.Disabled (not Dec_Check_Box.Checked);
   end On_Dec_Change;

begin
<...>
   Input_Text.Create (Fact_Form, 40);
   Input_Text.Place_Holder ("nombre entier");
   Input_Text.Pattern ("[0-9]+");
   Input_Text.Focus;
   Question_Label.Create (Fact_Form, Input_Text, "Saisir un nombre entier : ");
   Fact_Form.New_Line;
   Dec_Check_Box.Create (Fact_Form);
   Dec_Check_Box.On_Change_Handler (On_Dec_Change'Unrestricted_Access);
   Dec_Label.Create (Fact_Form, Dec_Check_Box, " Calcul décimal", False);
   Fact_Form.New_Line;
   Dec_Range.Create (Fact_Form);
   Dec_Range.Minimum ("0");
   Dec_Range.Maximum ("10");
   Dec_Range.Value (0);
   Dec_Range.Disabled;
   Dec_Value_Label.Create (Fact_Form, Dec_Range, "0", False);
<...>
   Fact_Button.On_Click_Handler (On_Factorial'Unrestricted_Access);
   Fact_Form.On_Submit_Handler (On_Factorial'Unrestricted_Access);
```

```
   Result_Label.Create (Main_View);
<...>
end hello;
```

L'affichage du curseur désactivé n'est pas très visible mais l'utilisateur ne peut plus le manipuler. Pour sa part, la contrainte de saisie n'est pas visible avec Safari, par contre, FireFox entoure la zone de saisie en rouge et affiche un message d'avertissement.

Le résultat, sur FireFox :



Source complet en annexe A en fin de document.

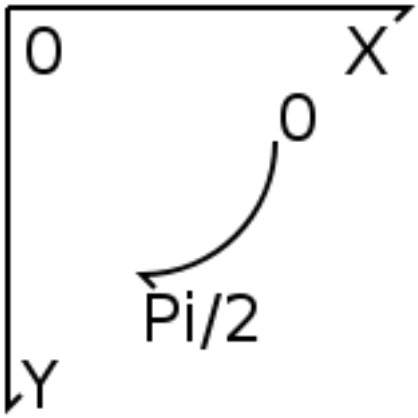
5. Les graphiques en 2D

L'utilisation des graphiques en 2D de GNOGA est un concept de création graphique qui demande de bien définir quelques mots de vocabulaire. Le premier est vectoriel, concept proche des mathématiques à base de cordonnées et de distances par opposition à bitmap collection de pixels allumés ou éteins à une position donnée. Le mode vectoriel est économe en ressource mémoire et permet des transformations du tracé sans perte de qualité alors que le mode bitmap est plus proche du fonctionnement de l'écran de notre ordinateur. En final, nous verrons que notre tracé vectoriel va s'appuyer sur un affichage bitmap pour être visible sur l'écran. Voyons les autres définitions des paquetages Gnoga.Gui.Element.Canvas et Gnoga.Gui.Element.Canvas.Context_2D.

GNOGA défini un contexte graphique représentant une surface plane cartésienne dont l'origine est en haut à gauche. Ce système de coordonnées est indépendant du dispositif d'affichage.

Ainsi, il reprend quelques concepts mathématiques :

- l'axe des abscisses (X ou horizontal) croissant va de la gauche vers la droite,
- l'axe des ordonnées (Y ou vertical) croissant va de haut en bas (au contraire de l'habitude en mathématiques),
- les coordonnées sont des nombres entiers,
- les deux axes se coupent à l'origine (0, 0),
- les angles croissants vont de l'axe X positif vers l'axe Y positif soit dans le sens horaire (au contraire de l'habitude en mathématiques). Ils sont donnés en degrés ou en radians soit la valeur de l'angle en degrés * PI / 180.0.



Chaque objet vectoriel (ligne, courbe, etc) est considéré comme un tracé dans le contexte graphique avant d'être transféré sur l'écran au moyen d'une surface de destination ou canvas.

Le canvas est créé dans une vue au moyen de la primitive Create avec la taille de la surface destination en pixel, par exemple :

```
Mon_Canvas.Create (Parent => Ma_Vue, Width  => 600, Height => 400);
```

Le contexte graphique est issu du canvas :

```
Context.Get_Drawing_Context_2D (Mon_Canvas);
```

Il est détruis lors de la destruction du canvas.

Les procédures de création d'un tracé sont plus communes :

- Begin_Path : commence un nouveau tracé au point origine (0, 0),
- Move_To : place le point courant aux coordonnées en paramètres,
- Line_To : trace une ligne droite vers un point donné en paramètre,
- Arc : trace un arc de cercle avec le centre, le rayon et les angles de départ et d'arrivée en paramètres,
- Rectangle : trace un rectangle avec le point en haut à gauche, la longueur et la largeur en paramètre.

GNOGA définit alors les procédures permettant de fixer le tracé sur la destination (le canvas) à partir du contexte :

- Stroke : transfert un tracé "fil de fer" avec la couleur et le style courants,
- Fill : transfert un tracé plein avec avec la couleur et le style courants,
- Stroke_Text : transfert un texte "fil de fer" avec la couleur, le style, la police et l'alignement courants.
- Fill_Text : transfert un texte plein avec la couleur, le style, la police et l'alignement courants.

Note : Stroke_Text et Fill_Text peuvent être combinés pour afficher un texte plein avec un contour différent.

Le contexte contient le style des lignes :

- Line_Width : définit la largeur du tracé,
- Set_Line_Dash : définit le motif du tracé (Empty_Dash_List : trait plein, Dotted_Dash_List : trait pointillé, Center_Dash_List : trait mixte, Dashed_Dash_List : trait avec tirets).

Le contexte contient le style des textes :

- Font : définit la police de caractères (comme avec CSS) avec sa taille, son style (Normal, Italic, Oblique), sa graisse (Weight_Normal, Weight_Bold),
- Text_Align : définit l'alignement du texte (Left : à gauche, Right : à droite, Center : centré, At_Start : au début suivant le sens lecture, To_End : à la fin suivant le sens lecture)

Le contexte contient les couleurs et les motifs d'affichage :

- Stroke_Color : définit la couleur du tracé en RGBA (Rouge, Vert, Bleu, Transparence) ou un nom de couleur (les principales couleurs sont dans l'unité Gnoga.Types.Colors),
- Fill_Color : définit de même la couleur de remplissage,
- Stroke_Pattern : définit un motif d'affichage du tracé,
- Fill_Pattern : définit un motif de remplissage du tracé.

Le context contient également une matrice de transformation par défaut qu'il est possible de modifier par :

- Translate : modifie la matrice par translation de l'origine avec les valeurs de distance en paramètre,
- Rotate : modifie la matrice par rotation des axes avec la valeur d'angle en paramètre,
- Scale : modifie la matrice par dilatation de l'échelle avec les valeurs de facteurs en paramètre (modifie également l'épaisseur des traits),
- Set_Transform (1.0, 0.0, 0.0, 1.0, 0.0, 0.0) : revient aux valeurs initiales (matrice identité)

L'application du tracé est conditionné par :

- Global_Alpha : définit le coefficient de transparence appliqué au tracé avant sa composition sur la destination,
- Glogal_Composite_Operation : définit l'opérateur de composition avec le paramètre prenant une valeur parmi :
  - Source_Over : applique la source sur la destination (mode par défaut),
  - Copy : applique la source en remplacement de la destination,
  - Xor_Copy : applique la source en effectuant un ou exclusif avec la destination.

  (beaucoup d'autres sont définis)

Le contenu du contexte peut être sauvegardé dans une pile :

- Save : sauvegarde le context courant dans la pile,
- Restore : restore le context courant depuis la pile.

En exemple le code produisant la figure au début du paragraphe :

```
Context.Translate (10, 10);
Context.Scale (5.0, 5.0);
Context.Font (Height => "5px");
Context.Move_To (0, 0);
Context.Line_To (30, 0);
Context.Line_To (28, 2);
Context.Fill_Text ("X", 24, 5);
Context.Move_To (0, 0);
Context.Line_To (0, 30);
Context.Line_To (2, 28);
Context.Fill_Text ("Y", 1, 26);
Context.Fill_Text ("0", 1, 5);
Context.Move_To (20, 10);
Context.Arc_Radians (10, 10, 10, 0.0, 3.14 / 2.0);
Context.Line_To (12, 22);
Context.Fill_Text ("0", 21, 10);
Context.Fill_Text ("Pi/2", 10, 27);
Context.Stroke;
```

Source complet en annexe B en fin de document.

6. Les évènements

Au paragraphe 4, nous avons vu qu'il était facile de déclencher une action avec un clic de souris sur un bouton ou récupérer une valeur du clavier avec une zone de saisie. Nous pouvons aussi facilement contrôler plus finement les évènements de la souris (clic ou mouvement) et ceux du clavier (appuyez ou relâcher une touche).

a) Typologie

Les évènements sont issus de l'interface utilisateur à travers des éléments visuels ou widgets. Le principe de base est de connecter une action (un sous-programme) avec un évènement. L'application réagit à cet évènement en appellant le sous-programme gestionnaire (handler) qui y est connecté. Bien que les évènements soient spécifiques pour chaque widget, la plupart sont définis dans Gnoga.Gui.Base. À nous de déterminer lesquels utiliser.

Voici quelques exemples les plus courants :

- Pour la souris relatif à un widget :
  - On_Click ou On_Mouse_Click : clic gauche sur un élément
  - On_Context_Menu ou On_Mouse_Right_Click : clic droit sur un élément pour ouvrir le menu contextuel
  - On_Double_Click ou On_Mouse_Double_Click : double clic gauche sur un élément
  - On_Mouse_Down : l'utilisateur à appuyer sur un bouton de la souris
  - On_Mouse_Up : l'utilisateur à relâcher un bouton de la souris
  - On_Mouse_Move : déplacement du pointeur de la souris au dessus de l'élément
- Pour le clavier relatif à une fenêtre :
  - On_Key_Down : l'utilisateur appuie sur une touche

- On_Key_Press, On_Character, On_Wide_Character : l'utilisateur a appuyé sur une touche d'un caractère (ou laisse appuyé)
- On_Key_Up : l'utilisateur relâche une touche

- Pour un formulaire relatif à un widget :
  - On_Blur : l'élément pert le focus
  - On_Focus : l'élément a le focus
  - On_Change : l'utilisateur a modifié le contenu de l'élément
  - On_Submit : l'utilisateur a cliqué sur le bouton de soumission du formulaire ou a appuyé sur entrée

Bien d'autres évènements sont disponibles, par exemple, pour le glisser-déposer ou les éléments multi-média, la liste complète est en fin de document à l'annexe C.

b) La spécification des gestionnaires (handlers)

Le gestionnaire (handler) prend une forme qui dépend de l'évènement :

- Dans le cas général :

```
procedure General (Object : in out Base_Type'Class);
```

- Pour la souris :

```
procedure Mouse (Object      : in out Base_Type'Class;
                 Mouse_Event : in     Mouse_Event_Record);
```

- Pour le clavier :

```
procedure Keyboard (Object         : in out Base_Type'Class;
                    Keyboard_Event : in     Keyboard_Event_Record);
```

- Pour un caractère :

```
procedure Char (Object : in out Base_Type'Class;
                Key    : in     Character);
```

- Pour un caractère étendu :

```
procedure Wide_Char (Object : in out Base_Type'Class;
                     Key    : in     Wide_Character);
```

c) La connexion

Nous connectons alors le widget émetteur de l'évènement et le gestionnaire (handler) définis avec la procédure de connexion qui reprend le nom de l'évènement suffixée par "_Handler", par exemple, pour un clic sur le bouton Quit_Button nous connectons l'évènement On_Click avec le gestionnaire (handler) On_Quit :

```
Quit_Button.On_Click_Handler (On_Quit'Access);
```

Un seul gestionnaire (handler) est connecté à un couple widget / évènement, si un deuxième est défini, il remplace alors le premier. Ce qui sert en autre pour annuler une connexion en passant la valeur null en lieu et place du gestionnaire (handler) :

```
Quit_Button.On_Click_Handler (null);
```

Les connexions seront détruites automatiquement lors de la destruction du widget.

d) Le code des gestionnaires (handlers)

Le code diffère suivant les paramètres à disposition.

Dans le cas général, il faut contraindre le type du paramètre Object avec celui du widget, par exemple pour un bouton :

```
procedure On_Click (Object : in out Gnoga.Gui.Base.Base_Type'Class) is
begin
  Gnoga.Log ("Visible = " & Button_Type (Object).Visible'Img);
  Gnoga.Log ("Height = " & Button_Type (Object).Height'Img);
  Gnoga.Log ("Width = " & Button_Type (Object).Width'Img);
end On_Click;
```

Dans le cas de la souris, nous avons à notre disposition toutes les informations utiles :

```
type Mouse_Message_Type is (Unknown, Click, Double_Click, Right_Click,
                            Mouse_Down, Mouse_Up, Mouse_Move);
type Mouse_Event_Record is
   record
     Message       : Mouse_Message_Type := Unknown;
     X             : Integer;
     Y             : Integer;
     Screen_X      : Integer;
     Screen_Y      : Integer;
     Left_Button   : Boolean := False;
     Middle_Button : Boolean := False;
     Right_Button  : Boolean := False;
     Alt           : Boolean := False;
     Control       : Boolean := False;
     Shift         : Boolean := False;
     Meta          : Boolean := False;
   end record;
```

Notes :

- Les positions X et Y sont relatives à la fenêtre et non au widget.
- Le point chaud du pointeur est au milieu de celui-ci et non en haut à gauche.
- Meta correspond à la touche Pomme ou Commande sur Mac.

Exemple de gestionnaire (handler) pour la souris :

```
procedure On_Mouse_Event
 (Object      : in out Gnoga.Gui.Base.Base_Type'Class;
  Mouse_Event :       Gnoga.Gui.Base.Mouse_Event_Record)   is
  pragma Unreferenced (Object);
  function Is_Button_Pressed
    (BN : Integer;
     ME : Gnoga.Gui.Base.Mouse_Event_Record) return Boolean     is
  begin
    case BN is
```

```
            when 1 =>  return ME.Left_Button;
            when 2 => return ME.Middle_Button;
            when 3 => return ME.Right_Button;
            when others => return False;
          end case;
        end Is_Button_Pressed;
      begin
        IntPositionX := Mouse_Event.X - Canvas.Offset_From_Left;
        IntPositionY := Mouse_Event.Y - Canvas.Offset_From_Top;
        if Mouse_Event.Message in Double_Click .. Mouse_Up then
          for BN in 1 .. 3 loop
            if Is_Button_Pressed (BN, Mouse_Event) and Mouse_Event.Message = Mouse_Down then
              IntButtons (BN).Pressed    := True;
              IntButtons (BN).LastXPress := IntPositionX;
              IntButtons (BN).LastYPress := IntPositionY;
              if IntButtons (BN).Press_Count < Integer'Last then
                IntButtons (BN).Press_Count := IntButtons (BN).Press_Count + 1;
              else
                IntButtons (BN).Press_Count := 0;
              end if;
            end if;
            if Is_Button_Pressed (BN, Mouse_Event) and Mouse_Event.Message = Mouse_Up then
              IntButtons (BN).Released     := True;
              IntButtons (BN).LastXRelease := IntPositionX;
              IntButtons (BN).LastYRelease := IntPositionY;
              if IntButtons (BN).Release_Count < Integer'Last then
                IntButtons (BN).Release_Count := IntButtons (BN).Release_Count + 1;
              else
                IntButtons (BN).Release_Count := 0;
              end if;
            end if;
            if Is_Button_Pressed (BN, Mouse_Event) and Mouse_Event.Message = Double_Click then
              IntButtons (BN).DoubleClic := True;
            end if;
          end loop;
        end if;
      end On_Mouse_Event;
```

Dans le cas du clavier, nous avons à notre disposition toutes les informations utiles (ou presque, voir notes) :

```
    type Keyboard_Message_Type is (Unknown, Key_Down, Key_Up, Key_Press);
    type Keyboard_Event_Record is
        record
          Message  : Keyboard_Message_Type := Unknown;
          Key_Code : Integer;
          Key_Char : Wide_Character;
          Alt      : Boolean := False;
          Control  : Boolean := False;
          Shift    : Boolean := False;
          Meta     : Boolean := False;
        end record;
```

Notes :

- Key_Code est un code de touche pour les évènements Key_Down et Key_Up,
- Key_Char est le caractère étendu pour l'évènement Key_Press dans le cas d'une touche représentant un caractère,
- Meta correspond à la touche Pomme ou Commande sur Mac,
- L'unité Gnoga.Types.Key_Code définit les constantes correspondantes aux codes des touches du clavier.

Exemple de gestionnaire (handler) pour le clavier :

```
      procedure On_Key_Press_Handler
        (Object         : in out Gnoga.Gui.Base.Base_Type'Class;
         Keyboard_Event : in     Gnoga.Gui.Base.Keyboard_Event_Record)
      is
         pragma Unreferenced (Object);
         Ch : constant Character := Ada.Characters.Conversions.To_Character
(Keyboard_Event.Key_Char);
         use type Gnoga.Gui.Base.Keyboard_Message_Type;
      begin
        if Ch /= Ada.Characters.Latin_1.NUL then
          Write_Key (Ch);
          return;
        end if;
        -- Other special keys
        if Keyboard_Event.Message = Gnoga.Gui.Base.Key_Down then
          case Keyboard_Event.Key_Code is
            when Gnoga.Types.Key_Codes.Key_BackSpace =>
              Write_Key (Ada.Characters.Latin_1.BS);
            when Gnoga.Types.Key_Codes.Key_Tab =>
              Write_Key (Ada.Characters.Latin_1.HT);
            when Gnoga.Types.Key_Codes.Key_F1 .. Gnoga.Types.Key_Codes.Key_F10 =>
              Write_Key (Ada.Characters.Latin_1.NUL);
              -- Alt modifier
              if Keyboard_Event.Alt then
                Write_Key
                  (Char'Val (Keyboard_Event.Key_Code - Gnoga.Types.Key_Codes.Key_F1 + 104));
              -- Control modifier
              elsif Keyboard_Event.Control then
                Write_Key
                  (Char'Val (Keyboard_Event.Key_Code - Gnoga.Types.Key_Codes.Key_F1 + 94));
              -- Shift modifier
              elsif Keyboard_Event.Shift then
                Write_Key
                  (Char'Val (Keyboard_Event.Key_Code - Gnoga.Types.Key_Codes.Key_F1 + 84));
              -- No modifier
              else
                Write_Key
                  (Char'Val (Keyboard_Event.Key_Code - Gnoga.Types.Key_Codes.Key_F1 + 59));
              end if;
            when Gnoga.Types.Key_Codes.Key_Home =>
              Write_Key (Ada.Characters.Latin_1.NUL);
              if Keyboard_Event.Control then
                Write_Key ('w');
              else
                Write_Key ('G');
```

```
      end if;
    when Gnoga.Types.Key_Codes.Key_Left =>
      Write_Key (Ada.Characters.Latin_1.NUL);
      if Keyboard_Event.Control then
        Write_Key ('s');
      else
        Write_Key ('K');
      end if;
    when Gnoga.Types.Key_Codes.Key_Up =>
      Write_Key (Ada.Characters.Latin_1.NUL);
      if Keyboard_Event.Control then
        Write_Key (Char'Val (141));
      else
        Write_Key ('H');
      end if;
    when Gnoga.Types.Key_Codes.Key_Right =>
      Write_Key (Ada.Characters.Latin_1.NUL);
      if Keyboard_Event.Control then
        Write_Key ('t');
      else
        Write_Key ('M');
      end if;
    when Gnoga.Types.Key_Codes.Key_Down =>
      Write_Key (Ada.Characters.Latin_1.NUL);
      if Keyboard_Event.Control then
        Write_Key (Char'Val (145));
      else
        Write_Key ('P');
      end if;
    when Gnoga.Types.Key_Codes.Key_Page_Up =>
      Write_Key (Ada.Characters.Latin_1.NUL);
      if Keyboard_Event.Control then
        Write_Key (Char'Val (132));
      else
        Write_Key ('I');
      end if;
    when Gnoga.Types.Key_Codes.Key_Page_Down =>
      Write_Key (Ada.Characters.Latin_1.NUL);
      if Keyboard_Event.Control then
        Write_Key ('v');
      else
        Write_Key ('Q');
      end if;
    when Gnoga.Types.Key_Codes.Key_End =>
      Write_Key (Ada.Characters.Latin_1.NUL);
      if Keyboard_Event.Control then
        Write_Key ('u');
      else
        Write_Key ('O');
      end if;
    when Gnoga.Types.Key_Codes.Key_Delete =>
      Write_Key (Ada.Characters.Latin_1.NUL);
      if Keyboard_Event.Control then
        Write_Key (Char'Val (147));
      else
        Write_Key ('S');
```

```
            end if;
         when others =>
            null;
      end case;
   end if;
end On_Key_Press_Handler;
```

7. Les éléments visuels de base

Une vue peut être agrémenter de plusieurs éléments visuels. Nous allons voir dans un premier temps les éléments de base définis dans le paquetage Gnoga.Gui.Element.Common.

Les éléments sont pour la plupart affichés en ligne à leur création, nous devrons alors ajouter des séparateurs comme New_Line pour les séparer. Notez que des possibilités d'agencement très sophistiquées sont possibles, à voir lors d'un prochain paragraphe sur les vues.

a) Typologie des éléments

- A_Type : affiche un lien Web, exemple :
Mon_Lien.Create (Main_View, "http://gnoga.com", "Gnoga");

- Button_Type : affiche un bouton, exemple :
Mon_Boutton.Create (Main_View, "OK");

- DIV_Type : affiche un élément bloc (identique à Put_Line), exemple :
Mon_Bloc.Create (Main_View, "En bloc.");

- P_Type : affiche un paragraphe, exemple :
Mon_Paragraphe.Create (Main_View, "En paragraphe.");

- IMG_Type : affiche une image, exemple :
Mon_Image.Create (Main_View, "favicon.ico");

- HR_Type : affiche une barre horizontale (identique à Horizontal_Rule), exemple :
Ma_Séparation.Create (Main_View);

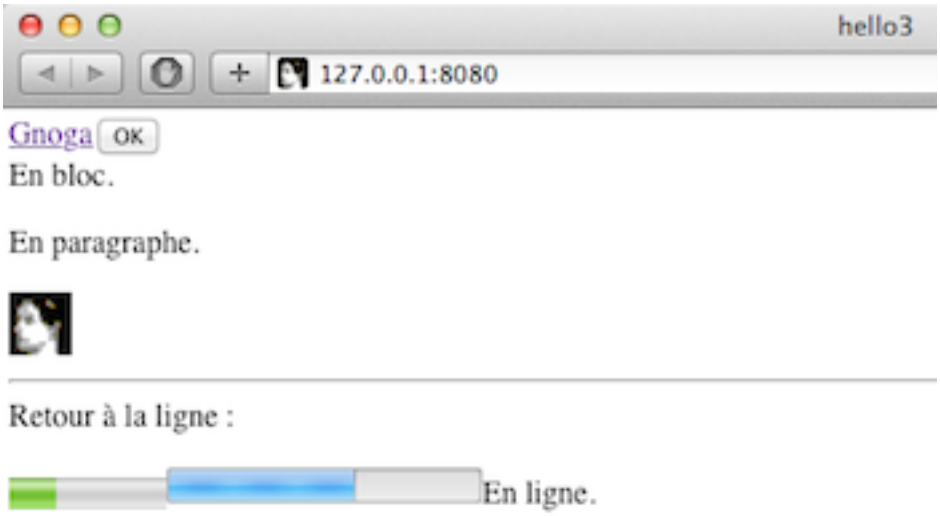
- BR_Type : affiche un retour à la ligne (identique à New_Line), exemple :
Mon_Retour.Create (Main_View);

- Meter_Type : affiche une jauge, exemple :
Ma_Jauge.Create (Main_View, 30);

- Progress_Bar_Type : affiche une barre de progression, exemple :
Ma_Barre.Create (Main_View, 60);

- Span_Type : affiche un élément en ligne (identique à Put), exemple :
Mon_Texte.Create (Main_View, "En ligne.");

Le résultat :



b) Les propriétés avancées

Les propriétés fixées explicitement ou par défaut à la création peuvent être lues ou modifiées avec les fonctions ou les procédures du même nom. Par exemple, la propriété Link de l'élément lien est lue avec la fonction :

function Link (A: A_Type) return String;

et est modifiée par la procédure :

procedure Link (A: in out A_Type; Value: String);

Les propriétés sont :

Éléments	Propriétés
A_Type	Link, Target
Button_Type	Disabled
Meter_Type	Value, High, Low, Maximum, Minimum, Optimum
Progress_Bar_Type	Value, Maximum

8. Les création de formulaires

Les éléments visuels disponibles pour un formulaire sont définis dans Gnoga.Gui.Element.Form. Ils se rapportent tous au formulaire qu'il faut créer auparavant, exemple :

```
Mon_Formulaire.Create (Main_View);
```

Les éléments sont pour la plupart affichés en ligne à leur création, nous devrons alors ajouter des séparateurs comme New_Line pour les séparer.

a) Typologie des éléments de formulaire

- Text_Area_Type : saisie d'un texte multi-ligne, exemple :
Mon_Texte_Multi.Create (Mon_Formulaire, Value => "Texte multi-ligne...");

- Hidden_Type : fixe une valeur dans un champ caché du formulaire, exemple :
Mon_Champ_Cache.Create (Mon_Formulaire, Value => "Valeur0", Name => "Champ0");

- Input_Button_Type : affiche un bouton, exemple :
Mon_Bouton.Create (Mon_Formulaire, "Annuler");

- Submit_Button_Type : affiche le bouton d'envoie du formulaire, exemple :
Mon_Bouton_Envoi.Create (Mon_Formulaire, "Envoyer");

- Reset_Button_Type : remet tous les champs à leur valeur initiale, exemple :
Mon_Bouton_RAZ.Create (Mon_Formulaire, "RAZ");

- Check_Box_Type : affiche une case à cocher, exemple :
Ma_Case_A_Cocher.Create (Mon_Formulaire);

- Radio_Button_Type : affiche un bouton de choix exclusif, exemple :
Mon_Bouton_Radio1.Create (Mon_Formulaire, Value => "Choix1", Name => "Choix");

- Input_Image_Type : affiche une image, les coordonnées de la position du pointeur de la souris lors du clic seront envoyées, exemple :
Mon_Image.Create (Mon_Formulaire, "favicon.ico");

- Text_Type : saisie d'une texte sur une ligne, exemple :
Mon_Texte.Create (Mon_Formulaire);

- Email_Type : saisie d'une adresse mel, exemple :
Mon_Mel.Create (Mon_Formulaire, Value => "mel@moi.org");

- Password_Type : saisie d'une mot de passe en aveugle (affiche des points), exemple :
Mon_Mot_Passe.Create (Mon_Formulaire, Value => "mdp");

- URL_Type : saisie d'une adresse web, exemple :
Mon_URL.Create (Mon_Formulaire, Value => "http://gnoga.com");

- Search_Type : saisie d'une recherche, exemple :
Ma_Recherche.Create (Mon_Formulaire, Value => "gnoga");

- Color_Picker_Type : saisie d'une couleur, exemple :
Ma_Couleur.Create (Mon_Formulaire);

- Date_Type : saisie d'une date (yyyy-mm-dd), exemple :
Ma_Date.Create (Mon_Formulaire);

- Time_Type : saisie d'une heure (HH:MM), exemple :
Mon_Heure.Create (Mon_Formulaire);

- Month_Type : saisie d'un mois (yyyy-mm), exemple :
Mon_Mois.Create (Mon_Formulaire);

- Week_Type : saisie d'une semaine (yyyy-Www), exemple :
Ma_Semaine.Create (Mon_Formulaire);

- Date_Time_Type : à ne plus utiliser, voir Date_Time_Local_Type.

- Date_Time_Local_Type : saisie de la date et de l'heure (yyyy-mm-ddTHH:MMZ), exemple :
Ma_Date_Heure_Locale.Create (Mon_Formulaire);

- Number_Type : saisie un nombre, exemple :
Mon_Nombre.Create (Mon_Formulaire, Value => "99");

- Range_Type : saisie avec un curseur, exemple :
Mon_Glisseur.Create (Mon_Formulaire);

- Label_Type : ajout d'une étiquette à un élément du formulaire, exemple :
Mon_Etiquette.Create (Mon_Formulaire, Mon_Glisseur, "Glisseur");

- Selection_Type : créé une sélection multiple, exemple :
Ma_Selection.Create (Mon_Formulaire, Name => "ChampSel");

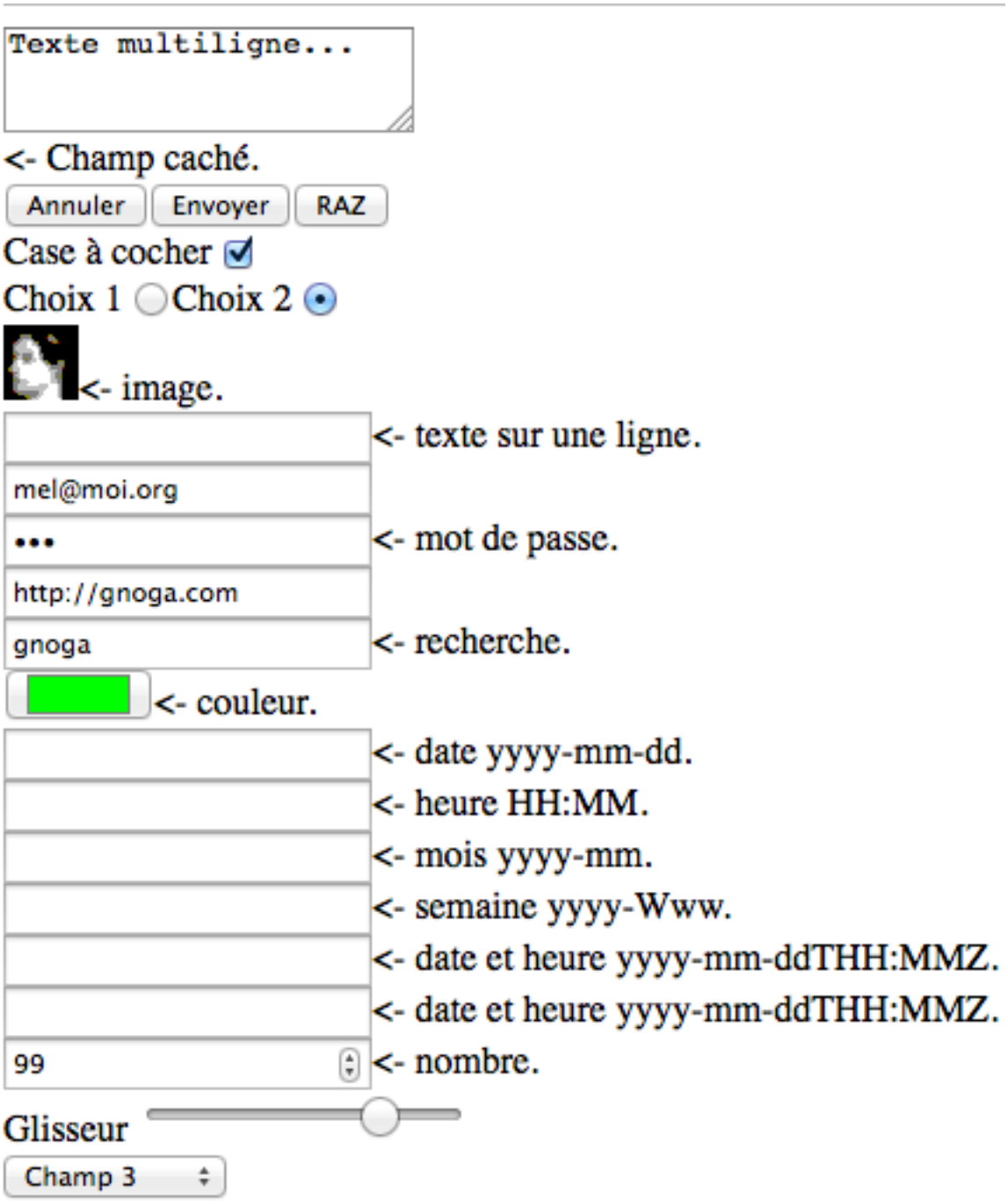
- Option_Type : ajout un élément à une sélection, exemple :
Mon_Option1.Create (Mon_Formulaire, Ma_Selection, "Valeur2", "Champ 2");

- Option_Group_Type : ajout d'un groupe à une sélection, exemple :
Mon_Option_Groupe.Create (Mon_Formulaire, Ma_Selection, "Groupe 1");

Tous les éléments de formulaire peuvent posséder une étiquette (même une étiquette), un nom et une valeur. Les noms et valeurs sont importants si le formulaire est envoyé via une requête GET ou POST, par exemple pour GET :

http://localhost:8080/?Champ0=Valeur0&Choix=Choix2&x=17&y=24&ChampSel=Valeur3

Le résultat :



Code source complet de hello3.adb en annexe D en fin de document.

b) Les propriétés avancées

Les propriétés fixées explicitement ou par défaut à la création peuvent être lues ou modifiées avec les fonctions ou les procédures du même nom. Par exemple, la propriété Name d'un élément est lue avec la fonction :

function Name (Element: Form_Element_Type) return String;

et est modifiée par la procédure :

procedure Name (Element: in out Form_Element_Type; Value: in String);

Les propriétés générales à tous les éléments sont :

- Autocomplete : complétion automatique ou non avec une liste fournie par l'élément Data_List_Type.
- Autofocus : focus automatique ou non lors de l'affichage du formulaire
- Disabled : activation ou non de l'élément
- Read_Only : modification possible ou non de l'élément
- Validate_On_Submit : validation ou non de l'élément lors de l'envoi
- Name : modification du nom du champ envoyé
- Default_Value : modification de la valeur par défaut de l'élément
- Value : valeur de l'élément
- Required : élément requis ou non lors de l'envoi du formulaire

Les propriétés particulières sont :

Éléments	Propriétés
Text_Area_Type	Word_Wrap, Columns, Rows
Check_Box_Type	Checked, Indeterminate
Radio_Button_Type	Checked
Input_Image_Type	Source
Text_Type	Pattern, Place_Holder, Size, Max_Length
Email_Type	Pattern, Place_Holder, Multiple_Emails
Password_Type	Pattern, Place_Holder
URL_Type	Pattern, Place_Holder
Search_Type	Pattern, Place_Holder
Color_Picker_Type	Color
Range_Type	Minimum, Maximum, Step
Label_Type	
Selection_Type	Multiple_Select, Visible_Lines, Selected, Text
Option_Type	Selected, Text
Option_Group_Type	Label

9. Les vues

Les vues au nombre de cinq sont à la base de tout affichage :
(de la plus simple à la plus complexe)
- Gnoga.Gui.View : vue de base
- Gnoga.Gui.View.Console : vue avec défilement automatique lors des ajouts
- Gnoga.Gui.View.Grid : vue sous forme de grilles
- Gnoga.Gui.View.Docker : vue avec éléments placés
- Gnoga.Gui.View.Card : vue avec éléments empilés

Une vue est un élément visuel comme un autre mais qui sert en plus de container pour les autres. C'est la raison pour laquelle en pratique la fenêtre de connexion a toujours une vue comme seul élément.

Une vue est un élément bloc qui peut être donc être placée dans une autre vue. Les éléments créés dans la vue sont placés comme décrit eu §7.

Les primitives communes à toutes les vues sont :
- Fill_Parent : provoque l'expansion de la vue pour remplir son parent
- Put_Line : affiche une chaine de caractère en mode bloc
- Put : affiche une chaine de caractère en ligne
- New_Line : affiche un retour à la ligne
- Horizontal_Line : affiche une ligne de séparation horizontale
- Put_HTML : ajoute du code HTML en ligne (le code doit être complet, c'est à dire commencer et finir par la même balise)

a) Gnoga.Gui.View

C'est la vue de base pour ajouter du texte et toute sorte d'éléments sans se préoccuper précisément de leur placement.

La primitive spécifique :
- Create : créé la vue dans son parent

Exemple d'un vue simple :

```
  Basic_View   : Gnoga.Gui.View.View_Type;
<...>
  Basic_View.Create (Parent_View);
  Basic_View.Background_Color (Gnoga.Types.Colors.Aqua);
  Basic_View.Put_Line ("Hello World! Basic");
```

b) Gnoga.Gui.View.Console

C'est la vue de base plus le défilement automatique lors de l'ajout des éléments.

La primitive spécifique :
- Create : créé la vue dans son parent

Exemple d'une vue console :

```
   Console_View : Gnoga.Gui.View.Console.Console_View_Type;
<...>
   Console_View.Create (Parent_View);
   Console_View.Background_Color (Gnoga.Types.Colors.Beige);
   Console_View.Put_Line ("Hello World! Console");
```

c) Gnoga.Gui.View.Grid

Cette vue permet de découper la vue parent en un pavage de plusieurs lignes et de plusieurs colonnes pour agencer nos éléments.

Les primitives spécifiques :

- Create : crée la vue dans son parent en définissant le pavage sous forme d'un tableau du nombre de lignes voulues avec des éléments COL pour une colonne simple ou SPN pour une extension de la colonne précédente, par exemple : ((COL, SPN), (COL, COL), (COL, SPN)) définit une ligne large de deux colonnes, une seconde ligne avec deux colonnes séparées et une troisième ligne large de deux colonnes
- Panel : retourne la vue de la ligne et de la colonne passés en paramètre

Exemple d'un pavage à une ligne et deux colonnes :

```
   Grid_View   : Gnoga.Gui.View.Grid.Grid_View_Type;
<...>
   Grid_View.Create (Parent_View, (1 => (COL, COL)), False);
   Grid_View.Background_Color (Gnoga.Types.Colors.Cyan);
   Grid_View.Panel (1, 1).Background_Color (Gnoga.Types.Colors.Dim_Gray);
   Grid_View.Panel (1, 1).Put_Line ("Hello World! Grid 1, 1");
   Grid_View.Panel (1, 2).Background_Color (Gnoga.Types.Colors.Forest_Green);
   Grid_View.Panel (1, 2).Put_Line ("Hello World! Grid 1, 2");
```

Contrairement à l'exemple ci-dessus, il est conseillé de placer une vue à utiliser dans chaque case plutôt que de l'utiliser la case directement. Nous aurons ainsi un placement plus fiable.

d) Gnoga.Gui.View.Docker

Cette vue permet de placer un élément sur un des quatre côtés plus le centre.

Les primitives spécifiques :

- Create : créé la vue dans son parent
- Left_Dock : ajoute un élément à gauche de la vue
- Right_Dock : ajoute un élément à droite de la vue
- Top_Dock : ajoute un élément en haut de la vue
- Bottom_Dock : ajoute un élément au bas de la vue
- Fill_Dock : ajoute un élément au centre de la vue
- On_Resize : permet d'adapter les dimensions de la vue lors d'un redimensionnement du parent car le placement est déterminé à la création de chaque élément

Exemple d'une vue contenant 5 éléments placés :

```
   DL, DR, DT, DB, DF : aliased Gnoga.Gui.Element.Common.P_Type;
```

```
<...>
   Docker_View.Create (Parent_View);
   Docker_View.Fill_Parent;
   DL.Create (Docker_View, "Hello World! Dock left");
   DL.Background_Color (Gnoga.Types.Colors.Light_Pink);
   Docker_View.Left_Dock (DL'Access);
   DR.Create (Docker_View, "Hello World! Dock right");
   DR.Background_Color (Gnoga.Types.Colors.Light_Coral);
   Docker_View.Right_Dock (DR'Access);
   DT.Create (Docker_View, "Hello World! Dock top");
   DT.Background_Color (Gnoga.Types.Colors.Light_Sea_Green);
   Docker_View.Top_Dock (DT'Access);
   DB.Create (Docker_View, "Hello World! Dock bottom");
   DB.Background_Color (Gnoga.Types.Colors.Light_Yellow);
   Docker_View.Bottom_Dock (DB'Access);
   DF.Create (Docker_View, "Hello World! Dock fill");
   DF.Background_Color (Gnoga.Types.Colors.Light_Gray);
   Docker_View.Fill_Dock (DF'Access);
```

Si la vue est créer directement dans la fenêtre de connexion alors il n'est pas nécessaire d'appeler "Fill_Parent" - contrairement à l'exemple ci-dessus.

e) Gnoga.Gui.View.Card

Cette vue permet de créé un empilement d'éléments; un seul est visible masquant les autres. Des onglets peuvent être ajoutés pour permettre à l'utilisateur de rendre visible l'élément correspondant à l'onglet.

Les primitives spécifiques :

- Create (Card) : créé la vue dans son parent
- Add_Card : ajoute un élément empilé sur les autres, il est identifié par le nom donné en paramètre, il est possible d'ajouter un élément vide
- Show_Card : affiche l'élément identifié par le nom donné en paramètre, les autres sont alors masqués
- On_Resize : permet d'adapter les dimensions de la vue lors d'un redimensionnement du parent car le placement est déterminé à la création de chaque élément
- Create (Tab) : créé un container pour onglets
- Add_Tab : ajouter un onglet identifié par le nom donné en paramètre associé à l'élément identifié par le nom donné en paramètre
- Select_Tab : affiche l'élément associé à l'onglet

Exemple d'une vue de deux éléments avec 2 onglets :

```
   Card_View.Create (Main_View.Panel (5, 1).all);
   Card_View.Fill_Parent;
   Tabs.Create (Card_View, Card_View);
   C1.Create (Card_View, "Hello World! C1");
   C1.Background_Color (To_String (Sky_Blue));
   Card_View.Add_Card ("C1", C1'Access, True);
   Tabs.Add_Tab ("C1", "Card 1");
   C2.Create (Card_View, "Hello World! C2");
   C2.Background_Color (To_String (Royal_Blue));
```

```
Card_View.Add_Card ("C2", C2'Access);
Tabs.Add_Tab ("C2", "Card 2");
```

Si la vue est créer directement dans la fenêtre de connexion alors il n'est pas nécessaire d'appeler "Fill_Parent" - contrairement à l'exemple ci-dessus.

f) Le résultat

Code source complet de hello4.adb en annexe E en fin de document.

10.Les résultats des formulaires

Suite au paragraphe "création de formulaires", nous allons voir comment récupérer les informations envoyées par l'utilisateur.

Il y a trois méthodes pour récupérer les informations d'un formulaire :

- la méthode GET : les données sont ajoutées à l'URI spécifiée dans le formulaire avec un encodage particulier, à utiliser principalement pour la consultation, la taille des données est limitée,
- la méthode POST : les données sont envoyées par une transaction spécifique, à utiliser pour la modification, la taille n'est pas limitée,
- la méthode des évènements : les données sont récupérées soit au changement d'un élément, au clic d'un bouton ou à la soumission du formulaire.

Les trois méthodes peuvent être utilisées en même temps.
Pour envoyer une parties des données avec la méthode GET et d'autres données avec la méthode POST, il nous faudra cependant créer au moins deux formulaires.

Reprenons l'exemple "hello3.adb" avec la méthode GET dans un premier temps, la méthode POST ensuite et enfin la méthode des évènements.

a) La méthode GET

Tout d'abord, je transforme le programme de Singleton en Multi_Connect car nous allons gérer deux pages, la page du formulaire et la page des résultats :

```
Gnoga.Application.Multi_Connect.Initialize;
Gnoga.Application.Multi_Connect.On_Connect_Handler (Formulaires'Unrestricted_Access);
Gnoga.Application.Multi_Connect.On_Connect_Handler
  (Resultats'Unrestricted_Access, "/resultats");
Gnoga.Application.Multi_Connect.Message_Loop;
```

Ensuite, j'ajoute la procédure qui va récupérer les informations soumises par le formulaire. Cette procédure va afficher pour chaque champ son nom ainsi que sa valeur. Je fais appel à la fonction Parse avec la valeur des paramètres et de leur encodage. Cette fonction retourne un container clé, valeur sur lequel je peux itérer :

```
procedure Resultats
  (Main_Window : in out Gnoga.Gui.Window.Window_Type'Class;
   Connection  :       access Gnoga.Application.Multi_Connect.Connection_Holder_Type)
is
   pragma Unreferenced (Connection);
   Main_View : Gnoga.Gui.View.Console.Console_View_Type;
begin
   Main_View.Create (Main_Window);
   if Main_Window.Location.Search /= "" then
      Main_View.Put_Line (Main_Window.Document.Input_Encoding);
      Main_View.Put_Line ("Get Results: " & Main_Window.Location.Search);
      for C in
        Gnoga.Gui.Location.Parse
          (Main_Window.Location.Search,
```

```
            Main_Window.Document.Input_Encoding).Iterate
         loop
           begin
             Main_View.Put_Line
             ("GET parameter: " &
              Gnoga.Types.Data_Maps.Key (C) &
              " = " &
              Gnoga.Types.Data_Maps.Element (C));
           end;
         end loop;
       end if;
     end Resultats;
```

Enfin, je référence cette page à la création du formulaire :

```
Mon_Formulaire.Create (Main_View, "/resultats");
```

Le résultat en cliquant sur le bouton "Envoyer" :

```
UTF-8
Get Results: ?Champ0=Valeur0&Choix=Choix2&ChampSel=Valeur1
GET parameter: ChampSel = Valeur1
GET parameter: Choix = Choix2
GET parameter: Champ0 = Valeur0
```

Le résultat en cliquant sur le l'image :

```
UTF-8
Get Results: ?Champ0=Valeur0&Choix=Choix2&ChampSel=Valeur1
GET parameter: ChampSel = Valeur1
GET parameter: Image.x = 29
GET parameter: Image.y = 26
GET parameter: Choix = Choix2
GET parameter: Champ0 = Valeur0
```

La position du clic sur l'image est envoyée. Les coordonnées sont préfixées du nom du champ image. En ajoutant une valeur à la création de l'image, la valeur sera présente si l'image est cliquée :

```
Mon_Image.Create (Mon_Formulaire, "favicon.ico", Value => "favicon.ico", Name => "Image");
```

Donne :

```
GET parameter: Image = favicon.ico
```

Seuls les champs ayant un nom sont retournés, j'ajoute donc un nom aux champs n'en ayant pas, ce qui a pour résultat :

```
UTF-8
Get Results: ?Texte+multi-ligne=1&Champ+Cach%C3%A9=Valeur0&Case+
%C3%A0+Cocher=&Boutons
+radio=Choix1&Texte=2&Mel=3&MDP=4&URL=5&Recheche=6&Couleur=7&Date=8&Heure=9&Mois=10
&Semaine=11&Date+heure+locale=12&Nombre=13&Glisseur=0&S%C3%A9lection=Valeur3
GET parameter: Couleur = 7
```

GET parameter: MDP = 4
GET parameter: Champ Caché = Valeur0
GET parameter: Mois = 10
GET parameter: Mel = 3
GET parameter: Date heure locale = 12
GET parameter: Heure = 9
GET parameter: Texte multi-ligne = 1
GET parameter: Nombre = 13
GET parameter: URL = 5
GET parameter: Boutons radio = Choix1
GET parameter: Case à Cocher =
GET parameter: Date = 8
GET parameter: Glisseur = 0
GET parameter: Recheche = 6
GET parameter: Texte = 2
GET parameter: Sélection = Valeur3
GET parameter: Semaine = 11

À noter que le champ de l'élément Check_Box_Type n'est présent avec une valeur vide que si la case est cochée sinon le champ n'est pas présent. En ajoutant une valeur à la création de la case à cocher, la valeur sera présente si la case est cochée :

```
Ma_Case_A_Cocher.Create (Mon_Formulaire, Value => "Cochée", Name => "Case à Cocher");
```

Donne :

GET parameter: Case à Cocher = Cochée

b) La méthode POST

Tout d'abord, comme pour la méthode GET, je transforme le programme de Singleton en Multi_Connect car nous allons gérer deux pages, la page du formulaire et la page des résultats :

```
Gnoga.Application.Multi_Connect.Initialize;
Gnoga.Application.Multi_Connect.On_Connect_Handler (Formulaires'Unrestricted_Access);
Gnoga.Application.Multi_Connect.On_Connect_Handler
  (Resultats'Unrestricted_Access, "/resultats");
Gnoga.Application.Multi_Connect.Message_Loop;
```

Ensuite, j'ajoute les procédures qui vont sélectionner les paramètres à récupérer et les sauvegarder :

```
procedure On_Post_Request
  (URI                : in     String;
   Accepted_Parameters :    out Ada.Strings.Unbounded.Unbounded_String)
is
   pragma Unreferenced (URI);
begin
   Accepted_Parameters :=
     Ada.Strings.Unbounded.To_Unbounded_String ("Texte multi-ligne,Champ Caché,Case à
Cocher,Boutons radio,Image,Texte,Mel,MDP,URL,Recherche,Couleur,Date,Heure,Mois,Semaine,Date
heure locale,Nombre,Glisseur,Sélection,Image.x,Image.y");
end On_Post_Request;
```

```
procedure On_Post (URI : String; Parameters : in out Gnoga.Types.Data_Map_Type) is
  pragma Unreferenced (URI);
begin
  Last_Parameters := Parameters;
end On_Post;
```

L'objet de sauvegarde des paramètres est déclaré juste au début du programme :

```
procedure hello3post is

  Last_Parameters : Gnoga.Types.Data_Map_Type;
```

Je référence ces procédures en les connectant au serveur :

```
  Gnoga.Application.Multi_Connect.Initialize;
  Gnoga.Application.Multi_Connect.On_Connect_Handler (Formulaires'Unrestricted_Access);
  Gnoga.Application.Multi_Connect.On_Connect_Handler
   (Resultats'Unrestricted_Access,
    "/resultats");

  Gnoga.Server.Connection.On_Post_Handler (On_Post'Unrestricted_Access);
  Gnoga.Server.Connection.On_Post_Request_Handler
   (On_Post_Request'Unrestricted_Access);

  Gnoga.Application.Multi_Connect.Message_Loop;
```

Je peux maintenant ajouter la procédure qui va récupérer les informations soumises par le formulaire. Cette procédure va afficher pour chaque champ son nom ainsi que sa valeur. Je fais aux paramètres sauvegardés sur lesquels je peux itérer :

```
procedure Resultats
  (Main_Window : in out Gnoga.Gui.Window.Window_Type'Class;
   Connection  :       access Gnoga.Application.Multi_Connect.Connection_Holder_Type)
is
  pragma Unreferenced (Connection);
  Main_View : Gnoga.Gui.View.Console.Console_View_Type;
begin
  Main_View.Create (Main_Window);
  for C in Last_Parameters.Iterate loop
    begin
      Main_View.Put_Line
      ("POST parameter: " &
       Gnoga.Types.Data_Maps.Key (C) &
       " = " &
       Gnoga.Types.Data_Maps.Element (C));
    end;
  end loop;
  Last_Parameters.Clear;
end Resultats;
```

La touche finale indispensable, modifier la création du formulaire en ajoutant la référence de la page de résultat et en indiquant la méthode Post :

```
begin
  Main_View.Create (Main_Window);
  Mon_Formulaire.Create (Main_View, "/resultats", Gnoga.Gui.Element.Form.Post);
```

Le résultat en cliquant sur le bouton "Envoyer" :

```
POST parameter: Recherche = 6
POST parameter: MDP = 4
POST parameter: Couleur = 7
POST parameter: Champ Caché =
POST parameter: Mois = 10
POST parameter: Mel = 3
POST parameter: Date heure locale = 12
POST parameter: Heure = 9
POST parameter: Texte multi-ligne = 1
POST parameter: Nombre = 13
POST parameter: URL = 5
POST parameter: Case à Cocher =
POST parameter: Boutons radio = Choix2
POST parameter: Image =
POST parameter: Date = 8
POST parameter: Glisseur = 50
POST parameter: Texte = 2
POST parameter: Sélection =
POST parameter: Semaine = 11
```

À noter : les paramètres "Champ Caché", "Case à Cocher", "Sélection" ne sont pas transmis car les caractères accentués ne sont pas pris en compte, je supprime donc les accents du champ Name et donne une valeur à la case à cocher et à l'image comme pour la méthode GET :

```
Mon_Champ_Cache.Create (Mon_Formulaire, Value => "Valeur0", Name => "Champ Cache");
Ma_Case_A_Cocher.Create (Mon_Formulaire, Value => "Cochée", Name => "Case a Cocher");
Mon_Image.Create (Mon_Formulaire, "favicon.ico", Value => "favicon.ico", Name => "Image");
Ma_Selection.Create (Mon_Formulaire, Name => "Selection");
```

Donne :

```
POST parameter: Champ Cache = Valeur0
POST parameter: Case a Cocher = Cochée
POST parameter: Image.x = 20
POST parameter: Image.y = 19
POST parameter: Image = favicon.ico
POST parameter: Selection = Valeur3
```

c) La méthode des évènements

Avec Gnoga, l'utilisation des évènements est la méthode la plus naturelle. C'est celle qui est recommandée. Nous allons déclencher un évènement à partir des boutons "Annuler" et "Envoyer". Les gestionnaires (handlers) correspondants doivent récupérer la valeurs des champs du formulaire. Pour cela, je mets la déclaration des champs du formulaire dans un objet dérivé de Connection_Data_Type.

```
type App_Data is new Gnoga.Types.Connection_Data_Type with record
```

```
      Main_View           : Gnoga.Gui.View.Console.Console_View_Type;
      Mon_Formulaire      : Gnoga.Gui.Element.Form.Form_Type;
      Mon_Texte_Multi     : Gnoga.Gui.Element.Form.Text_Area_Type;
      Mon_Champ_Cache     : Gnoga.Gui.Element.Form.Hidden_Type;
<...>
      Mon_Option2         : Gnoga.Gui.Element.Form.Option_Type;
      Mon_Option_Groupe   : Gnoga.Gui.Element.Form.Option_Group_Type;
    end record;
    type App_Access is access all App_Data;
```

Je modifie la procédure de création du formulaire en conséquence :

```
    procedure Formulaires (App : App_Access) is
    begin
      Main_Window.Connection_Data (App);
      App.Main_View.Create (Main_Window);
      App.Mon_Formulaire.Create (App.Main_View);
      App.Mon_Texte_Multi.Create (App.Mon_Formulaire, Value => "Texte multi-ligne...");
      App.Mon_Formulaire.New_Line;
      App.Mon_Champ_Cache.Create (App.Mon_Formulaire, Value => "Valeur0", Name =>
"Champ0");
      App.Mon_Formulaire.Put ("<- Champ caché.");
<...>
      App.Mon_Option1.Create (App.Mon_Formulaire, App.Ma_Selection, "Valeur2", "Champ 2");
      App.Mon_Option_Groupe.Create (App.Mon_Formulaire, App.Ma_Selection, "Groupe 1");
      App.Mon_Option2.Create (App.Mon_Formulaire, App.Mon_Option_Groupe, "Valeur3", "Champ
3");
    end Formulaires;
```

Ainsi que le programme principal :

```
    Gnoga.Application.Singleton.Initialize (Main_Window, Port => 8080);
    Formulaires (new App_Data);
    Gnoga.Application.Singleton.Message_Loop;
```

Nous pouvons alors ajouter les gestionnaires (handlers) des boutons "Annuler" et "Envoyer" en affichant les valeurs des champs avec la fonction "Value" pour la plus partet "Checked" pour la case à cocher et les boutons radio :

```
    procedure On_Cancel (Object : in out Gnoga.Gui.Base.Base_Type'Class) is
      App : constant App_Access := App_Access (Object.Connection_Data);
    begin
      App.Main_View.Put_HTML ("<h1>Annulation du formulaire</h1>");
    end On_Cancel;

    procedure On_Submit (Object : in out Gnoga.Gui.Base.Base_Type'Class) is
      App : constant App_Access := App_Access (Object.Connection_Data);
    begin
      App.Main_View.Put_Line ("EVT parameter: Mon_Texte_Multi = " & App.Mon_Texte_Multi.Value);
      App.Main_View.Put_Line ("EVT parameter: Mon_Champ_Cache = " &
App.Mon_Champ_Cache.Value);
      App.Main_View.Put_Line
      ("EVT parameter: Ma_Case_A_Cocher = " & App.Ma_Case_A_Cocher.Checked'Img);
```

```
      App.Main_View.Put_Line
      ("EVT parameter: Mon_Bouton_Radio1 = " & App.Mon_Bouton_Radio1.Checked'Img);
      App.Main_View.Put_Line
      ("EVT parameter: Mon_Bouton_Radio2 = " & App.Mon_Bouton_Radio2.Checked'Img);
      App.Main_View.Put_Line ("EVT parameter: Mon_Image = " & App.Mon_Image.Source);
      App.Main_View.Put_Line ("EVT parameter: Mon_Texte = " & App.Mon_Texte.Value);
<...>
      App.Main_View.Put_Line ("EVT parameter: Mon_Glisseur = " & App.Mon_Glisseur.Value);
      App.Main_View.Put_Line ("EVT parameter: Ma_Selection = " & App.Ma_Selection.Value);
    end On_Submit;
```

Le résultat en cliquant sur le bouton "Envoyer" puis "Annulation" :

```
EVT parameter: Mon_Texte_Multi = 1
EVT parameter: Mon_Champ_Cache = Valeur0
EVT parameter: Ma_Case_A_Cocher = TRUE
EVT parameter: Mon_Bouton_Radio1 = TRUE
EVT parameter: Mon_Bouton_Radio2 = FALSE
EVT parameter: Mon_Image = http://localhost:8080/favicon.ico
EVT parameter: Mon_Texte = 3
EVT parameter: Mon_Mel = 4
EVT parameter: Mon_Mot_Passe = 5
EVT parameter: Mon_URL = 6
EVT parameter: Ma_Recherche = 7
EVT parameter: Ma_Couleur = 8
EVT parameter: Ma_Date = 9
EVT parameter: Mon_Heure = 10
EVT parameter: Mon_Mois = 11
EVT parameter: Ma_Semaine = 12
EVT parameter: Ma_Date_Heure_Locale = 13
EVT parameter: Mon_Nombre = 14
EVT parameter: Mon_Glisseur = 47
EVT parameter: Ma_Selection = Valeur2

Annulation du formulaire
```

À noter que les caractères accentués ne sont pas rendus correctement.

Une autre façon de récupérer la valeur des champs du formulaire est de leur associer un gestionnaire (handler), exemple avec le glisseur :

```
    procedure On_Change (Object : in out Gnoga.Gui.Base.Base_Type'Class) is
      App : constant App_Access := App_Access (Object.Connection_Data);
    begin
      App.Main_View.Put_Line ("EVT change: Mon_Glisseur = " & App.Mon_Glisseur.Value);
    end On_Change;

    procedure Formulaires (App : App_Access) is
    begin
<...>
      App.Mon_Glisseur.Create (App.Mon_Formulaire);
      App.Mon_Glisseur.On_Change_Handler (On_Change'Unrestricted_Access);
<...>
    end Formulaires;
```

À chaque modification du curseur, sa valeur est affichée :

EVT change: Mon_Glisseur = 30

Code source complet de hello3get.adb, hello3post.adb et hello3evt.adb en annexe F en fin de document.

11.Les éléments visuels avancés

Après avoir vue les éléments visuels basiques, les graphiques, les formulaires, nous allons voir des éléments visuels composés comme :

- Gnoga.Gui.Element.List : les listes,
- Gnoga.Gui.Element.IFrame : les encapsulations de pages Web,
- Gnoga.Gui.Element.Multimedia : l'audio et la vidéo.

a) Les listes

Les listes peuvent être de plusieurs natures : les listes à puces, les listes ordonnées et les listes de définition.

Les listes sont par nature hiérarchiques. Nous créons d'abord la liste principale puis ses éléments puis les sous-listes que nous hiérarchisons puis leurs éléments, etc.

Les primitives de création :

- Create liste ou sous-liste : créer la liste ou la sous-liste dans son parent
- Create item : créer l'élément dans son parent avec le texte affiché

Les primitives communes aux listes :

- List_Kind : indique le type de puces (Disc - défaut, Circle, Square, None - rien) ou de numérotation (Armenian, Cjk_Ideographic, Decimal, Decimal_Leading_Zero, Georgian, Hebrew, Hiragana, Hiragana_Iroha, Katakana, Katakana_Iroha, Lower_Alpha, Lower_Greek, Lower_Latin, Lower_Roman, Upper_Alpha, Upper_Latin, Upper_Roman)
- List_Location : indique la position de la puce ou de la numérotation à l'intérieur ou à l'extérieur de l'élément HTML (Inside, Outside - défaut).
- Place_Inside_Bottom_Of : positionne une sous-liste à l'intérieur d'un élément de la liste principale

Exemple d'une liste à puce à deux niveaux :

```
   Unordered_List, Unordered_Sublist1,
  Unordered_Sublist2  : Gnoga.Gui.Element.List.Unordered_List_Type;
   ULItem1, ULItem2, ULItem3, ULItem11,
  ULItem21, ULItem22 : Gnoga.Gui.Element.List.List_Item_Type;
 <...>
   Unordered_List.Create (Main_View);

   ULItem1.Create (Unordered_List, "I1");

   Unordered_Sublist1.Create (Unordered_List);
   ULItem11.Create (Unordered_Sublist1, "SI11");
```

```
   ULItem2.Create (Unordered_List, "I2");

   Unordered_Sublist2.Create (Unordered_List);
   Unordered_Sublist2.List_Kind (Gnoga.Gui.Element.List.Square);

   ULItem21.Create (Unordered_Sublist2, "SI21");
   ULItem22.Create (Unordered_Sublist2, "SI22");

   ULItem3.Create (Unordered_List, "I3");
```

Exemple d'une liste hiérarchique ordonnée à deux niveaux :

```
   Ordered_List, Ordered_Sublist1,
 Ordered_Sublist2        : Gnoga.Gui.Element.List.Ordered_List_Type;
   OLItem1, OLItem2, OLItem3, OLItem11,
 OLItem21, OLItem22 : Gnoga.Gui.Element.List.List_Item_Type;
<...>
   Ordered_List.Create (Main_View);
   Ordered_List.List_Kind (Gnoga.Gui.Element.List.Decimal_Leading_Zero);

   OLItem1.Create (Ordered_List, "I1");
   OLItem2.Create (Ordered_List, "I2");
   OLItem3.Create (Ordered_List, "I3");

   Ordered_Sublist1.Create (Ordered_List);
   Ordered_Sublist1.Place_Inside_Bottom_Of (OLItem1);
   Ordered_Sublist2.Create (Ordered_List);
   Ordered_Sublist2.Place_Inside_Bottom_Of (OLItem2);
   Ordered_Sublist2.List_Location (Gnoga.Gui.Element.List.Inside);

   OLItem11.Create (Ordered_Sublist1, "SI11");
   OLItem21.Create (Ordered_Sublist2, "SI21");
   OLItem22.Create (Ordered_Sublist2, "SI22");
```

Exemple d'une liste de définition :

```
   Definition_List     : Gnoga.Gui.Element.List.Definition_List_Type;
   Term1, Term2, Term3 : Gnoga.Gui.Element.List.Term_Type;
   Desc1, Desc2, Desc3 : Gnoga.Gui.Element.List.Description_Type;
<...>
   Definition_List.Create (Main_View);

   Term1.Create (Definition_List, "I1");
   Desc1.Create (Definition_List, "-> D1");
   Term2.Create (Definition_List, "I2");
   Desc2.Create (Definition_List, "-> D2");
   Term3.Create (Definition_List, "I3");
   Desc3.Create (Definition_List, "-> D3");
```

Le résultat :

Unordered list

- I1
  - SI11
- I2
  - SI21
  - SI22
- I3

Ordered list

01. I1
    1. SI11
02. I2
    1. SI21
    2. SI22
03. I3

Description list

I1
: -> D1

I2
: -> D2

I3
: -> D3

b) Les encapsulations de pages Web

Cet élément permet d'afficher une page Web externe directement dans la page construite avec Gnoga.

La primitive de création :

- Create : crée dans le contenant en paramètre le cadre dans lequel va se placer la page Web en paramètre. (Note : le paramètre Seamless à True indiquant une apparence fondue dans le contenant n'est pas encore fonctionnel dans tous les navigateurs)

Exemple avec la page www.gnoga.com :

```
   Web_Frame : Gnoga.Gui.Element.IFrame.IFrame_Type;
<...>
   Main_View.Put_Line ("Gnoga web page (Internet connexion required):");
   Web_Frame.Create (Main_View, "http://www.gnoga.com");
   Web_Frame.Width (300);
   Web_Frame.Height (200);
   Web_Frame.Border;
```

Le résultat :

c) L'audio et la vidéo

Deux éléments nous sont proposés : un pour jouer un fichier audio et un autre pour jouer un fichier vidéo.

Noter que le fichier audio peut être un simple son bref et donc servir pour sonoriser une application.

Les primitives de création :

- Create audio : créer un élément audio avec la source en paramètre
- Create vidéo : créer un élément vidéo avec la source en paramètre

Les propriétés communes :

- Loop_Media : positionne ou retourne le mode de lecture en boucle,
- Media_Duration : retourne la durée en secondes,
- Media_Source : positionne ou retourne l'URL de la source,
- Media_Position : positionne ou retourne la position de la lecture en secondes,
- Muted : positionne ou retourne le mode silencieux,
- Paused : retourne True si le lecteur est en pause,
- Playback_Ended : retourne True si la lecture est finie,
- Playback_Rate : positionne ou retourne la vitesse de lecture de -1.0 à 1.0,
- Ready_To_Play : retourne True si le média est prêt à être joué,
- Seeking : retourne True si l'utilisateur manipule le lecteur,
- Volume : positionne ou retourne le niveau du volume sonore du lecteur.

Les primitives communes :

- Play : active la lecture du média,
- Pause : mais en pause la lecture du média,
- Load : charge ou recharge le média,
- Can_Play : retourne la possibilité de jouer le type de média en paramètre : video/ogg, video/mp4, video/webm, audio/mpeg, audio/ogg, audio/mp4, audio/mp3.

Les évènements communs :

- On_Media_Abort_Handler : le téléchargment a été interrompu,
- On_Media_Error_Handler : le chargement du média a provoqué une erreur,
- On_Can_Play_Handler : le média est prêt à être joué,
- On_Can_Play_Through_Handler : le média est prêt à être joué directement,
- On_Duration_Change_Handler : la durée complète du média est connue,
- On_Emptied_Handler : la liste de lecture est vide,
- On_Ended_Handler : la liste de lecture est terminée,
- On_Loaded_Data_Handler : le média est chargé,
- On_Loaded_Meta_Data_Handler : les méta-données du média sont chargées,
- On_Load_Start_Handler : le chargement du média démarre,
- On_Pause_Handler : l'utilisateur a mis le lecteur en pause,
- On_Play_Handler : l'utilisateur a mis le lecteur en lecture,
- On_Playing_Handler : le lecteur est en lecture,
- On_Progress_Handler : le téléchargement du média est en cours,
- On_Rate_Change_Handler : la vitesse de lecture a changée,
- On_Seeked_Handler : l'utilisateur a manipulé le lecteur,
- On_Seeking_Handler : l'utilisateur manipule le lecteur,

- On_Stalled_Handler : le média téléchargé est calé,
- On_Suspend_Handler : le téléchargement est fini ou en pause,
- On_Time_Update_Handler : la position de lecture a changée,
- On_Volume_Change_Handler : le volume de lecture a changé,
- On_Waiting_Handler : la vidéo s'est arrêté en attendant la prochaine trame à afficher.

Exemple de lecture d'un son et d'une vidéo :

```
   Audio : Gnoga.Gui.Element.Multimedia.Audio_Type;
   Video : Gnoga.Gui.Element.Multimedia.Video_Type;
<...>
  Main_View.Put_Line ("Audio");
   Audio.Create (Main_View, "img/tune.mp3");
   Gnoga.Log ("The browser can play MP3s? - " & Audio.Can_Play ("audio/mp3")'Img);

   Main_View.Put_Line ("Video");
   Video.Create (Main_View, "img/test.mp4");
   Gnoga.Log ("The browser can play MP4s? - " & Video.Can_Play ("video/mp4")'Img);
```

Le résultat :

Code source complet de hello5_list.adb, hello5_iframe.adb et hello5_av.adb en annexe G en fin de document.

12.Les connexions multiples

Pour l'instant, nous n'avons créer des programmes Singleton qui ne gère qu'une seule connexion avec le navigateur Web, à part le paragraphe sur les formulaires qui y fait exception. Nous faisions appel à Gnoga.Application.Singleton.Initialize pour attendre la connexion du navigateur Web. Une fois la connexion établie, le programme se déroule et attend la fin de la connexion avec Gnoga.Application.Singleton.Message_Loop puis se termine quand le navigateur met fin à la connexion. Ici, il n'y a qu'une seule connexion possible par exécution du programme.

Nous allons créer un petit programme qui affiche des clics de souris dans plusieurs fenêtres du navigateur. Pour cela nous allons invoquer le modèle "multi_connect", construire le programme suivant et l'exécuter :

```
$ cd <répertoire de test>
$ gnoga_make new hello6 multi_connect
# les répertoires css, html, img, js et src ont été créés
$ cd hello6
$ make
# les répertoires obj et bin ont été créés
$ bin/hello6
# une alerte peut apparaitre avec le message:
# "Voulez-vous que l'application « hello6 » accepte les connexions réseau entrantes ?"
# cliquer sans crainte sur le bouton "Refuser"
# ouvrez une fenêtre de navigation Web puis cliquez sur le bouton "Click Me"
# ouvrez une autre fenêtre de navigation Web puis cliquez également sur le bouton "Click Me"
# faites <crtl>-C pour arrêter le programme.
```

Analysons la structure du programme Hello6. En premier lieu, nous trouvons le programme principal (hello6.Main) avec les procédures d'initialisation et de fin de connexion mais ici pour les connexions multiples leur comportement est différent :

- Gnoga.Application.Multi_Connect.Initialize : initialise les connexions mais ne se met pas en attente,
- Gnoga.Application.Multi_Connect.Message_Loop : attend les connexions ou une fin de programme,
- Gnoga.Application.Multi_Connect.End_Application : provoque la fin du programme.

Comme une connexion est toujours possible, la fin du programme doit être explicite avec la procédure End_Application.

Une connexion a la forme d'un évènement auquel il faut définir un gestionnaire (handler). Nous pouvons définir un gestionnaire (handler) par défaut qui va répondre à toute connexion (il également possible de définir un gestionnaire (handler) pour une URL spécifique) :

```
Gnoga.Application.Multi_Connect.On_Connect_Handler (Default'Access, "default");
```

La procédure Default est du type :

```
type Application_Connect_Event is access
  procedure (Main_Window : in out Gnoga.Gui.Window.Window_Type'Class;
             Connection  : access Connection_Holder_Type);
```

Dans cette procédure Default, nous récupérons Main_Window pour créer notre page Web exactement de la même façon qu'avec le programme Singleton. Pourtant en le faisant de façon identique, les éléments visuels créés dans la page ne seront plus actifs. En fait la durée de vie d'un élément est liée à sa déclaration au sein de la procédure Default. À la fin de celle-ci, l'élément est désactivé mais pas supprimé de la page Web. Nous sommes confrontés à la persistance des éléments une fois la page créée.

Trois stratégies sont alors possibles : une simple, une classique et une orientée objet.

La première, simple, consiste à se mettre en attente de fin de connexion à la fin de notre procédure avec la procédure Hold. Nous revenons à un comportement identique à celui de notre programme Singleton :

```
procedure Default1
  (Main_Window : in out Gnoga.Gui.Window.Window_Type'Class;
   Connection  : access
     Gnoga.Application.Multi_Connect.Connection_Holder_Type)
is
  View : hello6.View.Default_View_Type;
begin
   View.Create (Main_Window);
   View.Click_Button.On_Click_Handler (On_Click1'Access);
  Connection.Hold;
end Default1;
```

Le fait que les éléments visuels restent locaux à notre procédure et ne sont donc pas accessibles en dehors est un inconvénient avec une gestion de formulaires, par exemple, par évènements mais pas forcément avec une gestion du formulaire avec GET ou POST.
Ici, une solution consiste à récupérer la vue comme parent de l'objet portant l'évènement :

```
procedure On_Click1 (Object : in out Gnoga.Gui.Base.Base_Type'Class) is
   View : hello6.View.Default_View_Access :=
          hello6.View.Default_View_Access (Object.Parent);
begin
   View.Label_Text.Put_Line ("Click1");
end On_Click1;
```

La deuxième stratégie, classique, consiste à mettre à profit Gnoga pour enregistrer les éléments visuels associées à la fenêtre de connexion :

```
type App_Data_Type is new Gnoga.Types.Connection_Data_Type with record
   View : hello6.View.Default_View_Type;
end record;
type App_Data_Access is access all App_Data_Type;

procedure Default2
  (Main_Window : in out Gnoga.Gui.Window.Window_Type'Class;
   Connection  : access
     Gnoga.Application.Multi_Connect.Connection_Holder_Type)
is
   App_Data : App_Data_Access := new App_Data_Type;
```

```
begin
   Main_Window.Connection_Data (App_Data);
   App_Data.View.Create (Main_Window);
   App_Data.View.Click_Button.On_Click_Handler (On_Click2'Access);
end Default2;
```

Les données sont récupérables dans un gestionnaire (handler) :

```
procedure On_Click2 (Object : in out Gnoga.Gui.Base.Base_Type'Class) is
begin
   App_Data_Access(Object.Connection_Data).View.Label_Text.Put_Line ("Click2");
end On_Click2;
```

Cette méthode est à privilégier avec Gnoga. Pour éviter les fuites de mémoires lors de connexions multiples, la mémoire allouée pour les données sera libérée automatiquement lors de la déconnexion. Attention, toutefois, la concurrence d'accès à ces données est laissé à l'entière responsabilité du programme (il n'y a pas de protection intrinsèque à Gnoga).

La troisième stratégie, dynamique, consiste définir les éléments visuels comme dynamiques avec une allocation de mémoire dynamique :

```
procedure Default3
  (Main_Window : in out Gnoga.Gui.Window.Window_Type'Class;
   Connection  : access
     Gnoga.Application.Multi_Connect.Connection_Holder_Type)
is
   View : hello6.View.Default_View_Access :=
     new hello6.View.Default_View_Type;
begin
   View.Dynamic;
   View.Create (Main_Window);
   View.Click_Button.On_Click_Handler (On_Click1'Access);
end Default3;
```

Pour éviter les fuites de mémoires lors de connexions multiples, l'appel à View.Dynamic enregistre View pour provoquer la libération de la mémoire allouée à View lors de la déconnexion.

L'ensemble des éléments visuels peut être déclarés ainsi dynamique ou seulement la vue qui contiendra alors les autres éléments par héritage de la vue standard.
C'est l'option qui a été prise dans la construction du model "multi_connect" :

```
type Default_View_Type is new Gnoga.Gui.View.View_Type with
   record
      Label_Text   : Gnoga.Gui.View.View_Type;
      Click_Button : Gnoga.Gui.Element.Common.Button_Type;
   end record;
type Default_View_Access is access all Default_View_Type;
```

Cette méthode est proche des modèles de programmation orientée objets tels qu'on les trouve avec Java.
L'accès à la vue depuis un gestionnaire (handler) est identique à la première méthode, la vue est récupérée comme parent de l'objet portant l'évènement :

```
procedure On_Click1 (Object : in out Gnoga.Gui.Base.Base_Type'Class) is
  View : hello6.View.Default_View_Access :=
         hello6.View.Default_View_Access (Object.Parent);
begin
  View.Label_Text.Put_Line ("Click1");
end On_Click1;
```

Suivant la construction Modèle-Vue-Contrôleur (MVC), les gestionnaires (handlers) d'évènements sont regroupés dans "hello6.Controller", la construction des éléments visuels dans "hello6.View". Il nous reste plus qu'à compléter notre programme en créant "hello6.Model" contenant l'accès aux données. Ce que nous verrons dans un prochain chapitre.

Astuce : avant de fermer une connexion depuis le programme, il faut au préalable détruire la vue principale pour libérer la mémoire des éléments de la page :

```
procedure On_Click3 (Object : in out Gnoga.Gui.Base.Base_Type'Class) is
  Window : Gnoga.Gui.Window.Window_Access :=
    Gnoga.Gui.Window.Window_Access (Object.Parent.Parent);
  View : Gnoga.Gui.View.View_Type;
begin
  Gnoga.Gui.View.View_Access(Window.Get_View).Remove;
  View.Create (Window.all);
  View.Put_Line ("Application exited.");
  Window.Close_Connection;
end On_Click3;
```

Notez que la fenêtre de connexion est obtenue comme parent de la vue, elle-même parent de l'objet portant l'évènement.

Une vue est créée localement pour informer l'utilisateur que sa connexion a été fermée.

Attention, la fermeture directe de la fenêtre de connexion avec la procédure Windows.Close n'est pas possible depuis le programme si celle-ci a été ouverte depuis le navigateur. Sinon nous obtenons une erreur du type :

Can't close the window since it was not opened by JavaScript

Des erreurs de connexions peuvent également apparaître ;

```
2017-05-04 11:55:14.77 : HTTP Server Started
2017-05-04 11:55:22.05 : New connection - ID 1
2017-05-04 11:55:25.12 : Websocket connection closed - ID 1
2017-05-04 11:55:25.12 : Deleting connection - 1
2017-05-04 11:55:45.34 : Swapping websocket connection  2 <=> 1
2017-05-04 11:55:45.34 : Connection error -  2
2017-05-04 11:55:45.34 : Old connection  1 already gone
2017-05-04 11:55:45.34 : Deleting connection - 2
2017-05-04 11:55:45.34 : Connection aborted -  2
2017-05-04 11:55:45.35 : Swapping websocket connection  3 <=> 1
2017-05-04 11:55:45.35 : Connection error -  3
2017-05-04 11:55:45.35 : Old connection  1 already gone
2017-05-04 11:55:45.35 : Deleting connection - 3
2017-05-04 11:55:45.35 : Connection aborted -  3
```

Cela est dû au mécanisme de reconnexion automatique de Gnoga en cas d'erreur ou de perte de connexion :

```
onerror: reconnect  boot.js:41:9
onclose: reconnect  boot.js:55:9
onerror: reconnect successful  boot.js:45:12
onclose: reconnect successful  boot.js:59:12
```

Code source complet de hello6* en annexe H en fin de document.

13.Le stockage de session

Le stockage de session permet de stocker des données (clé / valeur) associées à une session fenêtre ou onglet du navigateur. Pour mettre en oeuvre le stockage de session par le navigateur Web, nous allons réaliser une application multi-clients. Par soucis de rapidité je vais ici tout regrouper dans un même code source. Pour une application réelle, nous devrions créer une application avec connexions multiples comme vue au chapitre précédent.

Voici le code source de base, affichage d'un texte lors du clic du bouton :

```
procedure hello7_session is

  type App_Data_Type is new Gnoga.Types.Connection_Data_Type with record
     Main_View  : Gnoga.Gui.View.Console.Console_View_Type;
     Mon_Bouton : Gnoga.Gui.Element.Common.Button_Type;
  end record;
  type App_Data_Access is access all App_Data_Type;

  procedure On_Inc (Object : in out Gnoga.Gui.Base.Base_Type'Class) is
  begin
    App_Data_Access (Object.Connection_Data).Main_View.Put_Line ("Texte client : ");
  end On_Inc;

  procedure On_Connect
    (Main_Window : in out Gnoga.Gui.Window.Window_Type'Class;
     Connection  :        access Gnoga.Application.Multi_Connect.Connection_Holder_Type)
  is
     pragma Unreferenced (Connection);
     App_Data : constant App_Data_Access := new App_Data_Type;
  begin
     Main_Window.Connection_Data (App_Data);
     App_Data.Main_View.Create (Main_Window);
     App_Data.Mon_Bouton.Create (App_Data.Main_View, "Incrémenter du texte");
     App_Data.Mon_Bouton.On_Click_Handler (On_Inc'Unrestricted_Access);
    App_Data.Main_View.Put_Line ("Texte client : ");
  end On_Connect;

begin
  Gnoga.Application.Title ("hello7 session");
  Gnoga.Application.HTML_On_Close ("<b>Connection to Application has been terminated</b>");
```

```
      --    Gnoga.Application.Open_URL ("http://127.0.0.1:8080");
      Gnoga.Application.Multi_Connect.Initialize;
      Gnoga.Application.Multi_Connect.On_Connect_Handler (On_Connect'Unrestricted_Access);

      Gnoga.Application.Multi_Connect.Message_Loop;
    exception
      when E : others =>
        Gnoga.Log (Ada.Exceptions.Exception_Name (E) & " - " & Ada.Exceptions.Exception_Message
(E));
    end hello7_session;
```

Ouvrons plusieurs fenêtres ou onglets dans notre navigateur, le clic du bouton affiche toujours le même texte quelque soit la session fenêtre ou onglet. Nous allons alors enregistrer le texte dans le stockage de session du navigateur pour l'afficher de nouveau lors du clic qui ajoutera aussi un caractère aléatoire.

Les primitives de l'objet stockage de session sont :

- Session_Storage : retourne le stockage de session associé à l'objet passé en paramètre
- Length : retourne le nombre d'éléments stockés dans la session
- Key : retourne la clé à l'index passé en paramètre
- Set : positionne une valeur de stockage pour une clé passée en paramètre
- Get : retourne la valeur d'une clé passée en paramètre ("null" si la clé n'existe pas)
- Script_Accessor : retourne l'identification du stockage ("sessionStorage")

Nous ajoutons un objet stockage de session associé à la fenêtre dans nos données de connexion :

```
  type App_Data_Type is new Gnoga.Types.Connection_Data_Type with record
    Main_View  : Gnoga.Gui.View.Console.Console_View_Type;
    Mon_Bouton : Gnoga.Gui.Element.Common.Button_Type;
    Session    : Gnoga.Client.Storage.Session_Storage_Type;
  end record;
```

Nous initialisons le stockage de session avec la clé "TEXT" à la connexion la fenêtre :

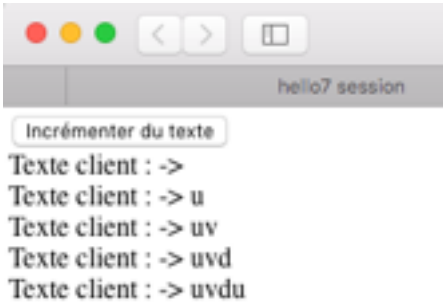
```
  begin
    Main_Window.Connection_Data (App_Data);
    App_Data.Main_View.Create (Main_Window);
    App_Data.Mon_Bouton.Create (App_Data.Main_View, "Incrémenter du texte");
    App_Data.Mon_Bouton.On_Click_Handler (On_Inc'Unrestricted_Access);
    App_Data.Session := Gnoga.Client.Storage.Session_Storage (Main_Window);
    App_Data.Session.Set ("TEXT", " -> ");
    Char_Random.Reset (Char_Generator);
    App_Data.Main_View.Put_Line ("Texte client : " & App_Data.Session.Get ("TEXT"));
  end On_Connect;
```

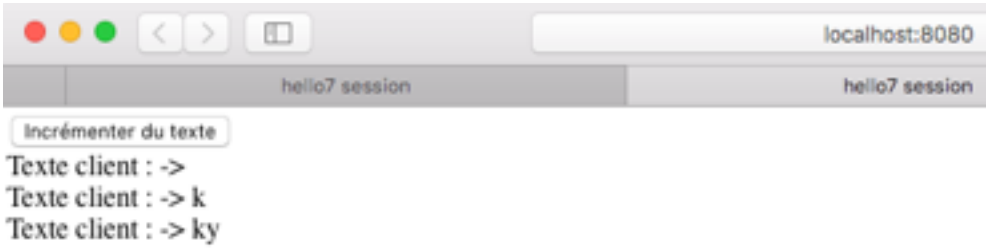
Finalement nous ajoutons un caractère aléatoire lors du clic sur le bouton :

```
  procedure On_Inc (Object : in out Gnoga.Gui.Base.Base_Type'Class) is
    Client_Text : constant String :=
     App_Data_Access (Object.Connection_Data).Session.Get ("TEXT") &
     Char_Random.Random (Char_Generator);
```

```
   begin
      App_Data_Access (Object.Connection_Data).Session.Set ("TEXT", Client_Text);
      App_Data_Access (Object.Connection_Data).Main_View.Put_Line ("Texte client : " &
Client_Text);
   end On_Inc;
```

Le résultat :





Le texte qui s'incrémente d'un caractère est propre à chaque session, ici deux onglets.
Le texte de l'onglet de droite est indépendant de celui de gauche.

Note : le stockage de session ne fonctionnent pas avec les fenêtres privées de Safari.

Gnoga apporte une fonctionnalité supplémentaire avec la possibilité de stocker un identifiant unique par session avec la fonction Gnoga_Session_ID. Celle-ci retourne l'identifiant de la clé donnée en paramètre ("gid" par défaut) si elle existe sinon un identifiant unique est créé et stocké dans la session avec la clé passée en paramètre.

Nous allons utilisé cet identifiant comme clé d'un container avec la valeur du caractère à ajouter au texte. Comme l'identifiant est unique, chaque session aura son caractère sauvegardé du côté serveur.

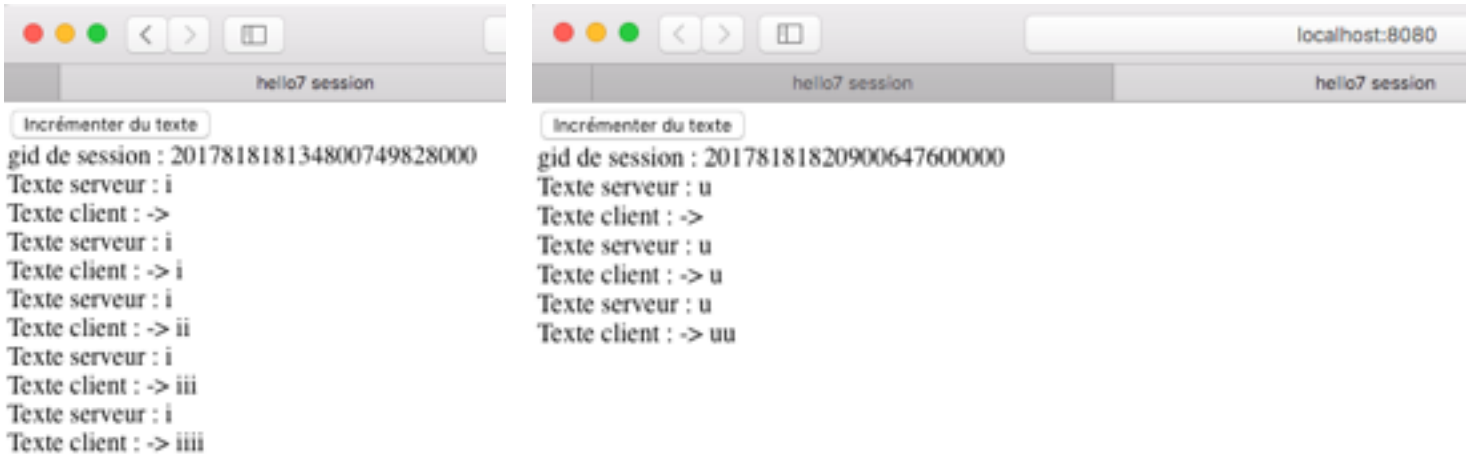
Nous ajoutons l'objet container avec l'identifiant unique et le caractère à ajouter :

```
   begin
      Main_Window.Connection_Data (App_Data);
      App_Data.Main_View.Create (Main_Window);
      App_Data.Mon_Bouton.Create (App_Data.Main_View, "Incrémenter du texte");
      App_Data.Mon_Bouton.On_Click_Handler (On_Inc'Unrestricted_Access);
      App_Data.Main_View.Put_Line ("gid de session : " & Main_Window.Gnoga_Session_ID);
      App_Data.Session := Gnoga.Client.Storage.Session_Storage (Main_Window);
      App_Data.Session.Set ("TEXT", " -> ");
      Char_Random.Reset (Char_Generator);
      Text_Map.Include (Main_Window.Gnoga_Session_ID, Char_Random.Random
(Char_Generator));
      App_Data.Main_View.Put_Line ("Texte serveur : " & Text_Map.Element
(Main_Window.Gnoga_Session_ID));
      App_Data.Main_View.Put_Line ("Texte client : " & App_Data.Session.Get ("TEXT"));
   end On_Connect;
```

Nous ajoutons le caractère propre à chaque session lors du clic sur le bouton :

```
   procedure On_Inc (Object : in out Gnoga.Gui.Base.Base_Type'Class) is
      Client_Text : constant String :=
        App_Data_Access (Object.Connection_Data).Session.Get ("TEXT") &
        Text_Map.Element (App_Data_Access (Object.Connection_Data).Session.Get ("gid"));
    begin
      App_Data_Access (Object.Connection_Data).Session.Set ("TEXT", Client_Text);
      App_Data_Access (Object.Connection_Data).Main_View.Put_Line
      ("Texte serveur : " & Text_Map.Element (App_Data_Access
(Object.Connection_Data).Session.Get ("gid")));
      App_Data_Access (Object.Connection_Data).Main_View.Put_Line ("Texte client : " &
Client_Text);
    end On_Inc;
```

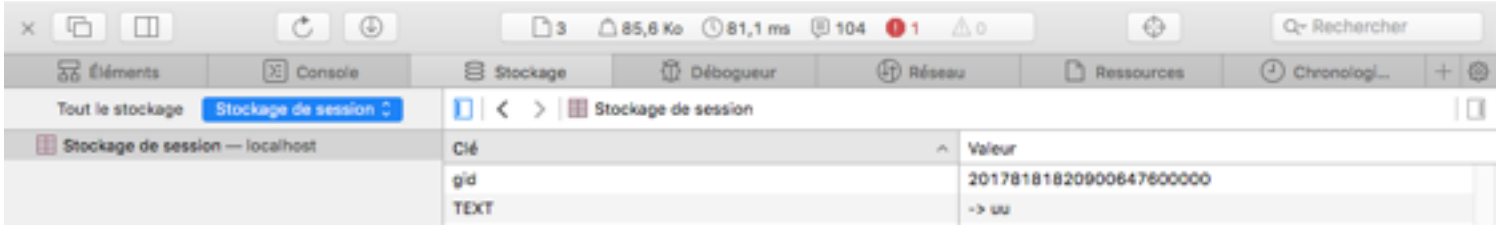
Le résultat :



Le texte qui s'incrémente d'un caractère est propre à chaque session, ici deux onglets.
Le texte de l'onglet de droite est indépendant de celui de gauche.
Le caractère qui s'ajoute est propre à chaque session.

Code source complet de hello7_session en annexe I en fin de document.

Les bidouilleurs noteront qu'il est facile de flouer le serveur en attribuant une autre valeur de clé dans le stockage de session avec l'inspecteur du navigateur, par exemple :



Un mécanise plus élaboré devra alors mis en place.

14.Le stockage local

Le stockage local permet de stocker des données (clé / valeur) associées à une URL par le navigateur. Pour mettre en oeuvre le stockage local par le navigateur Web, nous allons réaliser une application multi-clients. Par soucis de rapidité je vais reprendre ici le code de base du chapitre précédent.

Ajoutons une zone de saisie de texte par l'utilisateur ainsi que son affichage par le bouton "Envoyer" (simule son envoi) :

```
procedure hello7_local is

   type App_Data_Type is new Gnoga.Types.Connection_Data_Type with record
      Main_View       : Gnoga.Gui.View.Console.Console_View_Type;
      Mon_Bouton      : Gnoga.Gui.Element.Common.Button_Type;
      Mon_Formulaire  : Gnoga.Gui.Element.Form.Form_Type;
      Mon_Texte_Multi : Gnoga.Gui.Element.Form.Text_Area_Type;
   end record;
   type App_Data_Access is access all App_Data_Type;

   procedure On_Envoie (Object : in out Gnoga.Gui.Base.Base_Type'Class) is
   begin
      App_Data_Access (Object.Connection_Data).Main_View.Put_Line
      ("Texte envoyé : " & App_Data_Access
(Object.Connection_Data).Mon_Texte_Multi.Value);
   end On_Envoie;

   procedure On_Connect
     (Main_Window : in out Gnoga.Gui.Window.Window_Type'Class;
      Connection  :        access Gnoga.Application.Multi_Connect.Connection_Holder_Type)
   is
      pragma Unreferenced (Connection);
      App_Data : constant App_Data_Access := new App_Data_Type;
   begin
      Main_Window.Connection_Data (App_Data);
      App_Data.Main_View.Create (Main_Window);
      App_Data.Mon_Bouton.Create (App_Data.Main_View, "Envoyer du texte");
      App_Data.Mon_Bouton.On_Click_Handler (On_Envoie'Unrestricted_Access);
      App_Data.Mon_Formulaire.Create (App_Data.Main_View);
      App_Data.Mon_Texte_Multi.Create (App_Data.Mon_Formulaire);
      App_Data.Mon_Texte_Multi.Focus;
      App_Data.Main_View.Put_Line ("Texte envoyé : ");
   end On_Connect;
```

Ouvrons une fenêtre, ajoutons du texte dans la zone de saisie, si nous fermons la fenêtre, si nous rechargeons la page ou si nous ouvrons une autre connexion alors le texte déjà saisi est perdu.

Nous allons alors enregistrer le texte dans le stockage local du navigateur pour l'afficher de nouveau lors de l'ouverture d'une nouvelle page. Le texte n'est plus perdu même lors d'une perte de connexion ou mauvaise action de l'utilisateur.

Les primitives de l'objet stockage local sont :

- Local_Storage : retourne le stockage local associé à l'objet passé en paramètre
- Length : retourne le nombre d'éléments stockés dans le navigateur
- Key : retourne la clé à l'index passé en paramètre
- Set : positionne une valeur de stockage pour une clé passée en paramètre
- Get : retourne la valeur d'une clé passée en paramètre ("null" si la clé n'existe pas)
- Script_Accessor : retourne l'identification du stockage ("localStorage")

Nous ajoutons un objet stockage local associé à la fenêtre dans nos données de connexion :

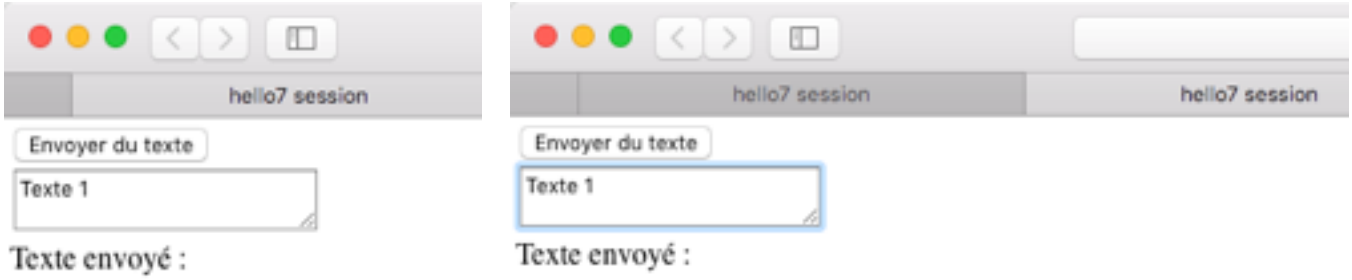
```
type App_Data_Type is new Gnoga.Types.Connection_Data_Type with record
  Main_View       : Gnoga.Gui.View.Console.Console_View_Type;
  Mon_Bouton      : Gnoga.Gui.Element.Common.Button_Type;
  Mon_Formulaire  : Gnoga.Gui.Element.Form.Form_Type;
  Mon_Texte_Multi : Gnoga.Gui.Element.Form.Text_Area_Type;
  Local           : Gnoga.Client.Storage.Local_Storage_Type;
end record;
```

Nous créons le gestionnaire de modification du stockage local, l'objet de stockage local, initialisons la zone de texte avec la valeur de la clé "TEXT" du stockage local et créons finalement le gestionnaire de modification du texte saisi :

```
begin
  Main_Window.Connection_Data (App_Data);
  Main_Window.On_Storage_Handler (On_Modif_Stockage'Unrestricted_Access);
  App_Data.Local := Gnoga.Client.Storage.Local_Storage (Main_Window);
  App_Data.Main_View.Create (Main_Window);
  App_Data.Mon_Bouton.Create (App_Data.Main_View, "Envoyer du texte");
  App_Data.Mon_Bouton.On_Click_Handler (On_Envoie'Unrestricted_Access);
  App_Data.Mon_Formulaire.Create (App_Data.Main_View);
  App_Data.Mon_Texte_Multi.Create (App_Data.Mon_Formulaire, Value =>
App_Data.Local.Get ("TEXT"));
  App_Data.Mon_Texte_Multi.Focus;
  App_Data.Mon_Texte_Multi.On_Change_Handler (On_Modif_Texte'Unrestricted_Access);
  App_Data.Main_View.Put_Line ("Texte envoyé : ");
end On_Connect;
```

Nous créons le code des gestionnaires de saisie,

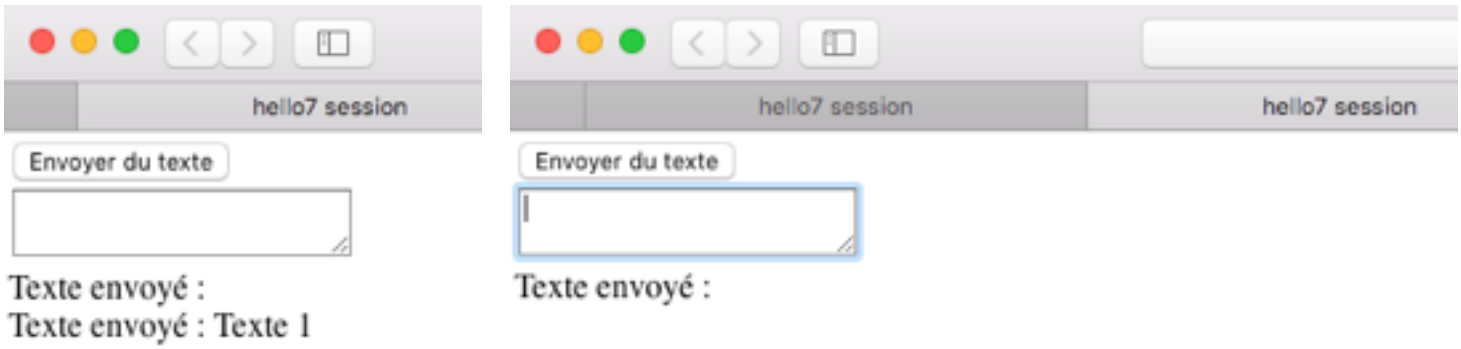
Le résultat :



Le texte de l'onglet de droite est présent automatiquement dans l'onglet de gauche.

Si le navigateur est quitté puis relancé, le texte sera conservé prêt a être modifié et à être envoyé.

Note : le stockage local ne fonctionnent pas avec les fenêtres privées de Safari.



Le texte a été envoyé avec l'onglet de gauche et a alors disparu dans l'onglet de droite.

Pascal Pignard, décembre 2014, janvier-septembre 2015, avril-août 2017.

Annexe A : le code complet de hello.adb

```
with Ada.Exceptions;
with Ada.Long_Long_Float_Text_IO;

with Gnoga.Application.Singleton;
with Gnoga.Gui.Window;
with Gnoga.Gui.View;
with Gnoga.Gui.Base;
with Gnoga.Gui.Element.Common;
with Gnoga.Gui.Element.Form;

procedure hello is
  Main_Window     : Gnoga.Gui.Window.Window_Type;
  Main_View       : Gnoga.Gui.View.View_Type;
  Quit_Button     : Gnoga.Gui.Element.Common.Button_Type;
  Fact_Form       : Gnoga.Gui.Element.Form.Form_Type;
  Input_Text      : Gnoga.Gui.Element.Form.Text_Type;
  Question_Label  : Gnoga.Gui.Element.Form.Label_Type;
  Fact_Button     : Gnoga.Gui.Element.Common.Button_Type;
  Result_Label    : Gnoga.Gui.Element.Common.DIV_Type;
  Dec_Check_Box   : Gnoga.Gui.Element.Form.Check_Box_Type;
  Dec_Label       : Gnoga.Gui.Element.Form.Label_Type;
  Dec_Range       : Gnoga.Gui.Element.Form.Range_Type;
  Dec_Range_Label : Gnoga.Gui.Element.Form.Label_Type;
  Dec_Value_Label : Gnoga.Gui.Element.Form.Label_Type;

  procedure On_Quit (Object : in out Gnoga.Gui.Base.Base_Type'Class) is
    pragma Unreferenced (Object);
  begin
    Gnoga.Application.Singleton.End_Application;
  end On_Quit;

  procedure On_Factorial (Object : in out Gnoga.Gui.Base.Base_Type'Class) is
    pragma Unreferenced (Object);
    function Factorielle (N : Natural) return Long_Long_Integer is
      F : Long_Long_Integer := 1;
    begin
      for I in 2 .. N loop
        F := F * Long_Long_Integer (I);
      end loop;
      return F;
    end Factorielle;
    function Factorielle (N : Natural) return String is
      F : Long_Long_Float := 1.0;
      S : String (1 .. 20);
    begin
      for I in 2 .. N loop
        F := F * Long_Long_Float (I);
      end loop;
      Ada.Long_Long_Float_Text_IO.Put (S, F, Dec_Range.Value, 6);
      return S;
    end Factorielle;
  begin
    if not Dec_Check_Box.Checked then
      Result_Label.Text
```

```
         ("Résultat " &
          Input_Text.Value &
          "! = " &
          Long_Long_Integer'Image
            (Factorielle (Natural'Value (Input_Text.Value))));
      else
         Result_Label.Text
         ("Résultat " &
          Input_Text.Value &
          "! = " &
          Factorielle (Natural'Value (Input_Text.Value)));
      end if;
   exception
      when others =>
         Result_Label.Text ("Erreur !");
   end On_Factorial;

   procedure Dec_Range_Change
     (Element : in out Gnoga.Gui.Base.Base_Type'Class)
   is
      pragma Unreferenced (Element);
   begin
      Dec_Value_Label.Text (Dec_Range.Value);
   end Dec_Range_Change;

   procedure On_Dec_Change (Element : in out Gnoga.Gui.Base.Base_Type'Class) is
      pragma Unreferenced (Element);
   begin
      Dec_Range.Disabled (not Dec_Check_Box.Checked);
   end On_Dec_Change;

begin
   Gnoga.Application.Title ("hello");
   Gnoga.Application.HTML_On_Close
     ("<b>Connection to Application has been terminated</b>");

   Gnoga.Application.Open_URL ("http://127.0.0.1:8081");
   Gnoga.Application.Singleton.Initialize (Main_Window, Port => 8081);

   Main_View.Create (Main_Window);
   Quit_Button.Create (Main_View, "Quitter");
   Quit_Button.On_Click_Handler (On_Quit'Unrestricted_Access);
   Fact_Form.Create (Main_View);
   Input_Text.Create (Fact_Form, 40);
   Input_Text.Place_Holder ("nombre entier");
   Input_Text.Pattern ("[0-9]+");
   Input_Text.Focus;
   Question_Label.Create (Fact_Form, Input_Text, "Saisir un nombre entier : ");
   Fact_Form.New_Line;
   Dec_Check_Box.Create (Fact_Form);
   Dec_Check_Box.On_Change_Handler (On_Dec_Change'Unrestricted_Access);
   Dec_Label.Create (Fact_Form, Dec_Check_Box, " Calcul décimal", False);
   Fact_Form.New_Line;
   Dec_Range.Create (Fact_Form);
   Dec_Range.Minimum (0);
   Dec_Range.Maximum (10);
```

```
   Dec_Range.Value (0);
   Dec_Range.Disabled;
   Dec_Value_Label.Create (Fact_Form, Dec_Range, "0", False);
   Dec_Range_Label.Create (Fact_Form, Dec_Range, " décimale(s)", False);
   Dec_Range.On_Change_Handler (Dec_Range_Change'Unrestricted_Access);
   Main_View.Put_Line ("Calcul de factorielle :");
   Fact_Button.Create (Main_View, "Calcul");
   Fact_Button.On_Click_Handler (On_Factorial'Unrestricted_Access);
   Fact_Form.On_Submit_Handler (On_Factorial'Unrestricted_Access);
   Result_Label.Create (Main_View);

   Gnoga.Application.Singleton.Message_Loop;
exception
   when E : others =>
      Gnoga.Log
        (Ada.Exceptions.Exception_Name (E) &
         " - " &
         Ada.Exceptions.Exception_Message (E));
end hello;
```

Annexe B : le code complet de hello2.adb

```
with Ada.Exceptions;

with Gnoga.Application.Singleton;
with Gnoga.Gui.Window;
with Gnoga.Gui.View;
with Gnoga.Gui.Element.Canvas;
with Gnoga.Gui.Element.Canvas.Context_2D;

procedure hello2 is
  Main_Window : Gnoga.Gui.Window.Window_Type;
  Main_View   : Gnoga.Gui.View.View_Type;
  Mon_Canvas  : Gnoga.Gui.Element.Canvas.Canvas_Type;
  Context     : Gnoga.Gui.Element.Canvas.Context_2D.Context_2D_Type;
begin
  Gnoga.Application.Title ("hello2");
  Gnoga.Application.HTML_On_Close
    ("<b>Connection to Application has been terminated</b>");

  Gnoga.Application.Open_URL ("http://127.0.0.1:8080");
  Gnoga.Application.Singleton.Initialize (Main_Window, Port => 8080);

  Main_View.Create (Main_Window);

  Mon_Canvas.Create (Parent => Main_View, Width => 600, Height => 400);
  Context.Get_Drawing_Context_2D (Mon_Canvas);
  Context.Translate (10, 10);
  Context.Scale (5.0, 5.0);
  Context.Font (Height => "5px");
  Context.Move_To (0, 0);
  Context.Line_To (30, 0);
  Context.Line_To (28, 2);
  Context.Fill_Text ("X", 24, 5);
  Context.Move_To (0, 0);
  Context.Line_To (0, 30);
  Context.Line_To (2, 28);
  Context.Fill_Text ("Y", 1, 26);
  Context.Fill_Text ("0", 1, 5);
  Context.Move_To (20, 10);
  Context.Arc_Radians (10, 10, 10, 0.0, 3.14 / 2.0);
  Context.Line_To (12, 22);
  Context.Fill_Text ("0", 21, 10);
  Context.Fill_Text ("Pi/2", 10, 27);
  Context.Stroke;

  Gnoga.Application.Singleton.Message_Loop;
exception
  when E : others =>
    Gnoga.Log
      (Ada.Exceptions.Exception_Name (E) &
       " - " &
       Ada.Exceptions.Exception_Message (E));
end hello2;
```

Annexe C : les événements de Gnoga par paquetage

Connexion	Paquetage
On_Connect_Handler	gnoga-application-multi_connect.ads:74
On_Resize_Handler	gnoga-gui-base.ads:345
On_Scroll_Handler	gnoga-gui-base.ads:350
On_Focus_Handler	gnoga-gui-base.ads:357
On_Blur_Handler	gnoga-gui-base.ads:362
On_Change_Handler	gnoga-gui-base.ads:367
On_Focus_In_Handler	gnoga-gui-base.ads:371
On_Focus_Out_Handler	gnoga-gui-base.ads:375
On_Input_Handler	gnoga-gui-base.ads:379
On_Reset_Handler	gnoga-gui-base.ads:383
On_Search_Handler	gnoga-gui-base.ads:390
On_Select_Handler	gnoga-gui-base.ads:394
On_Submit_Handler	gnoga-gui-base.ads:398
On_Click_Handler	gnoga-gui-base.ads:407
On_Mouse_Click_Handler	gnoga-gui-base.ads:412
On_Context_Menu_Handler	gnoga-gui-base.ads:418
On_Mouse_Right_Click_Handler	gnoga-gui-base.ads:423
On_Double_Click_Handler	gnoga-gui-base.ads:429
On_Mouse_Double_Click_Handler	gnoga-gui-base.ads:434
On_Mouse_Enter_Handler	gnoga-gui-base.ads:440
On_Mouse_Leave_Handler	gnoga-gui-base.ads:445
On_Mouse_Over_Handler	gnoga-gui-base.ads:450
On_Mouse_Out_Handler	gnoga-gui-base.ads:455
On_Mouse_Down_Handler	gnoga-gui-base.ads:460
On_Mouse_Up_Handler	gnoga-gui-base.ads:466
On_Mouse_Move_Handler	gnoga-gui-base.ads:472
On_Drag_Start_Handler	gnoga-gui-base.ads:480
On_Drag_Handler	gnoga-gui-base.ads:489
On_Drag_End_Handler	gnoga-gui-base.ads:493
On_Drag_Enter_Handler	gnoga-gui-base.ads:497
On_Drag_Leave_Handler	gnoga-gui-base.ads:501
On_Drop_Handler	gnoga-gui-base.ads:505
On_Character_Handler	gnoga-gui-base.ads:516

Connexion	Paquetage
On_Wide_Character_Handler	gnoga-gui-base.ads:521
On_Key_Down_Handler	gnoga-gui-base.ads:526
On_Key_Up_Handler	gnoga-gui-base.ads:531
On_Key_Press_Handler	gnoga-gui-base.ads:536
On_Copy_Handler	gnoga-gui-base.ads:543
On_Cut_Handler	gnoga-gui-base.ads:547
On_Paste_Handler	gnoga-gui-base.ads:551
On_Create_Handler	gnoga-gui-base.ads:557
On_Destroy_Handler	gnoga-gui-base.ads:563
On_Child_Added_Handler	gnoga-gui-base.ads:572
On_Child_Removed_Handler	gnoga-gui-base.ads:577
On_Message_Handler	gnoga-gui-base.ads:582
On_Resize	gnoga-gui-base.ads:601
On_Create	gnoga-gui-base.ads:604
On_Destroy	gnoga-gui-base.ads:608
On_Child_Added	gnoga-gui-base.ads:612
On_Child_Removed	gnoga-gui-base.ads:616
On_Message	gnoga-gui-base.ads:621
On_Media_Abort_Handler	gnoga-gui-element-multimedia.ads:139
On_Media_Error_Handler	gnoga-gui-element-multimedia.ads5
On_Can_Play_Handler	gnoga-gui-element-multimedia.ads:151
On_Can_Play_Through_Handler	gnoga-gui-element-multimedia.ads:157
On_Duration_Change_Handler	gnoga-gui-element-multimedia.ads:164
On_Emptied_Handler	gnoga-gui-element-multimedia.ads:170
On_Ended_Handler	gnoga-gui-element-multimedia.ads:176
On_Loaded_Data_Handler	gnoga-gui-element-multimedia.ads:182
On_Loaded_Meta_Data_Handler	gnoga-gui-element-multimedia.ads:188
On_Load_Start_Handler	gnoga-gui-element-multimedia.ads:194
On_Pause_Handler	gnoga-gui-element-multimedia.ads:200
On_Play_Handler	gnoga-gui-element-multimedia.ads:205
On_Playing_Handler	gnoga-gui-element-multimedia.ads:210
On_Progress_Handler	gnoga-gui-element-multimedia.ads:215
On_Rate_Change_Handler	gnoga-gui-element-multimedia.ads:221
On_Seeked_Handler	gnoga-gui-element-multimedia.ads:227
On_Seeking_Handler	gnoga-gui-element-multimedia.ads:233

Connexion	Paquetage
On_Stalled_Handler	gnoga-gui-element-multimedia.ads:239
On_Suspend_Handler	gnoga-gui-element-multimedia.ads:245
On_Time_Update_Handler	gnoga-gui-element-multimedia.ads:251
On_Volume_Change_Handler	gnoga-gui-element-multimedia.ads:257
On_Waiting_Handler	gnoga-gui-element-multimedia.ads:262
On_Message	gnoga-gui-element-multimedia.ads:273
On_Resize	gnoga-gui-plugin-ace_editor.ads:157
On_Open_Handler	gnoga-gui-plugin-jqueryui-widget.ads6
On_Close_Handler	gnoga-gui-plugin-jqueryui-widget.ads:150
On_Message	gnoga-gui-plugin-jqueryui-widget.ads:164
On_Resize	gnoga-gui-view-card.ads:100
On_Child_Added	gnoga-gui-view-console.ads:66
On_Resize	gnoga-gui-view-docker.ads:119
On_Child_Added	gnoga-gui-view.ads:187
On_Abort_Handler	gnoga-gui-window.ads:270
On_Error_Handler	gnoga-gui-window.ads:274
On_Before_Unload_Handler	gnoga-gui-window.ads:278
On_Hash_Change_Handler	gnoga-gui-window.ads:283
On_Orientation_Change_Handler	gnoga-gui-window.ads:288
On_Storage_Handler	gnoga-gui-window.ads:293
On_Resize	gnoga-gui-window.ads:304
On_Child_Added	gnoga-gui-window.ads:309
On_Message	gnoga-gui-window.ads:314
On_Connect_Handler	gnoga-server-connection.ads:111
On_Post_Request_Handler	gnoga-server-connection.ads:134
On_Post_Handler	gnoga-server-connection.ads3
On_Post_File_Handler	gnoga-server-connection.ads:152

Annexe D : le code complet de hello3.adb

```
with Ada.Exceptions;

with Gnoga.Application.Singleton;
with Gnoga.Gui.Window;
with Gnoga.Gui.View.Console;
with Gnoga.Gui.Element.Common;
with Gnoga.Gui.Element.Form;

procedure hello3 is
  Main_Window : Gnoga.Gui.Window.Window_Type;
  Main_View   : Gnoga.Gui.View.Console.Console_View_Type;

  procedure Elements_Basics is
    Mon_Lien       : Gnoga.Gui.Element.Common.A_Type;
    Mon_Boutton    : Gnoga.Gui.Element.Common.Button_Type;
    Mon_Bloc       : Gnoga.Gui.Element.Common.DIV_Type;
    Mon_Paragraphe : Gnoga.Gui.Element.Common.P_Type;
    Mon_Image      : Gnoga.Gui.Element.Common.IMG_Type;
    Ma_Séparation  : Gnoga.Gui.Element.Common.HR_Type;
    Mon_Retour     : Gnoga.Gui.Element.Common.BR_Type;
    Ma_Jauge       : Gnoga.Gui.Element.Common.Meter_Type;
    Ma_Barre       : Gnoga.Gui.Element.Common.Progress_Bar_Type;
    Mon_Texte      : Gnoga.Gui.Element.Common.Span_Type;
  begin
    Mon_Lien.Create (Main_View, "http://gnoga.com", "Gnoga");
    Mon_Boutton.Create (Main_View, "OK");
    Mon_Bloc.Create (Main_View, "En bloc.");
    Mon_Paragraphe.Create (Main_View, "En paragraphe.");
    Mon_Image.Create (Main_View, "favicon.ico");
    Ma_Séparation.Create (Main_View);
    Main_View.Put ("Retour à la ligne :");
    Mon_Retour.Create (Main_View);
    Ma_Jauge.Create (Main_View, 30);
    Ma_Barre.Create (Main_View, 60);
    Mon_Texte.Create (Main_View, "En ligne.");
  end Elements_Basics;

  procedure Formulaires is
    Mon_Formulaire       : Gnoga.Gui.Element.Form.Form_Type;
    Mon_Texte_Multi      : Gnoga.Gui.Element.Form.Text_Area_Type;
    Mon_Champ_Cache      : Gnoga.Gui.Element.Form.Hidden_Type;
    Mon_Bouton           : Gnoga.Gui.Element.Form.Input_Button_Type;
    Mon_Bouton_Envoi     : Gnoga.Gui.Element.Form.Submit_Button_Type;
    Mon_Bouton_RAZ       : Gnoga.Gui.Element.Form.Reset_Button_Type;
    Ma_Case_A_Cocher     : Gnoga.Gui.Element.Form.Check_Box_Type;
    Mon_Bouton_Radio1    : Gnoga.Gui.Element.Form.Radio_Button_Type;
    Mon_Bouton_Radio2    : Gnoga.Gui.Element.Form.Radio_Button_Type;
    Mon_Image            : Gnoga.Gui.Element.Form.Input_Image_Type;
    Mon_Texte            : Gnoga.Gui.Element.Form.Text_Type;
    Mon_Mel              : Gnoga.Gui.Element.Form.Email_Type;
    Mon_Mot_Passe        : Gnoga.Gui.Element.Form.Password_Type;
    Mon_URL              : Gnoga.Gui.Element.Form.URL_Type;
    Ma_Recherche         : Gnoga.Gui.Element.Form.Search_Type;
    Ma_Couleur           : Gnoga.Gui.Element.Form.Color_Picker_Type;
```

```
   Ma_Date             : Gnoga.Gui.Element.Form.Date_Type;
   Mon_Heure           : Gnoga.Gui.Element.Form.Time_Type;
   Mon_Mois            : Gnoga.Gui.Element.Form.Month_Type;
   Ma_Semaine          : Gnoga.Gui.Element.Form.Week_Type;
   Ma_Date_Heure_Locale : Gnoga.Gui.Element.Form.Date_Time_Local_Type;
   Mon_Nombre          : Gnoga.Gui.Element.Form.Number_Type;
   Mon_Glisseur        : Gnoga.Gui.Element.Form.Range_Type;
   Mon_Etiquette       : Gnoga.Gui.Element.Form.Label_Type;
   Ma_Selection        : Gnoga.Gui.Element.Form.Selection_Type;
   Mon_Option1         : Gnoga.Gui.Element.Form.Option_Type;
   Mon_Option2         : Gnoga.Gui.Element.Form.Option_Type;
   Mon_Option_Groupe   : Gnoga.Gui.Element.Form.Option_Group_Type;
begin
   Mon_Formulaire.Create (Main_View);
   Mon_Texte_Multi.Create (Mon_Formulaire, Value => "Texte multi-ligne...");
   Mon_Formulaire.New_Line;
   Mon_Champ_Cache.Create (Mon_Formulaire, Value => "Valeur0", Name => "Champ0");
   Mon_Formulaire.Put ("<- Champ caché.");
   Mon_Formulaire.New_Line;
   Mon_Bouton.Create (Mon_Formulaire, "Annuler");
   Mon_Bouton_Envoi.Create (Mon_Formulaire, "Envoyer");
   Mon_Bouton_RAZ.Create (Mon_Formulaire, "RAZ");
   Mon_Formulaire.New_Line;
   Ma_Case_A_Cocher.Create (Mon_Formulaire);
   Gnoga.Gui.Element.Form.Label_Access'(new Gnoga.Gui.Element.Form.Label_Type).Create
   (Mon_Formulaire, Ma_Case_A_Cocher, "Case à cocher");
   Mon_Formulaire.New_Line;
   Mon_Bouton_Radio1.Create (Mon_Formulaire, Value => "Choix1", Name => "Choix");
   Gnoga.Gui.Element.Form.Label_Access'(new Gnoga.Gui.Element.Form.Label_Type).Create
   (Mon_Formulaire, Mon_Bouton_Radio1, "Choix 1");
   Mon_Bouton_Radio2.Create (Mon_Formulaire, True, Value => "Choix2", Name => "Choix");
   Gnoga.Gui.Element.Form.Label_Access'(new Gnoga.Gui.Element.Form.Label_Type).Create
   (Mon_Formulaire, Mon_Bouton_Radio2, "Choix 2");
   Mon_Formulaire.New_Line;
   Mon_Image.Create (Mon_Formulaire, "favicon.ico");
   Mon_Formulaire.Put ("<- image.");
   Mon_Formulaire.New_Line;
   Mon_Texte.Create (Mon_Formulaire);
   Mon_Formulaire.Put ("<- texte sur une ligne.");
   Mon_Formulaire.New_Line;
   Mon_Mel.Create (Mon_Formulaire, Value => "mel@moi.org");
   Mon_Formulaire.New_Line;
   Mon_Mot_Passe.Create (Mon_Formulaire, Value => "mdp");
   Mon_Formulaire.Put ("<- mot de passe.");
   Mon_Formulaire.New_Line;
   Mon_URL.Create (Mon_Formulaire, Value => "http://gnoga.com");
   Mon_Formulaire.New_Line;
   Ma_Recherche.Create (Mon_Formulaire, Value => "gnoga");
   Mon_Formulaire.Put ("<- recherche.");
   Mon_Formulaire.New_Line;
   Ma_Couleur.Create (Mon_Formulaire);
   Mon_Formulaire.Put ("<- couleur.");
   Mon_Formulaire.New_Line;
   Ma_Date.Create (Mon_Formulaire);
   Mon_Formulaire.Put ("<- date yyyy-mm-dd.");
   Mon_Formulaire.New_Line;
```

```
      Mon_Heure.Create (Mon_Formulaire);
      Mon_Formulaire.Put ("<- heure HH:MM.");
      Mon_Formulaire.New_Line;
      Mon_Mois.Create (Mon_Formulaire);
      Mon_Formulaire.Put ("<- mois yyyy-mm.");
      Mon_Formulaire.New_Line;
      Ma_Semaine.Create (Mon_Formulaire);
      Mon_Formulaire.Put ("<- semaine yyyy-Www.");
      Mon_Formulaire.New_Line;
      Ma_Date_Heure_Locale.Create (Mon_Formulaire);
      Mon_Formulaire.Put ("<- date et heure locale yyyy-mm-ddTHH:MMZ.");
      Mon_Formulaire.New_Line;
      Mon_Nombre.Create (Mon_Formulaire, Value => "99");
      Mon_Formulaire.Put ("<- nombre.");
      Mon_Formulaire.New_Line;
      Mon_Glisseur.Create (Mon_Formulaire);
      Mon_Etiquette.Create (Mon_Formulaire, Mon_Glisseur, "Glisseur");
      Mon_Formulaire.New_Line;
      Ma_Selection.Create (Mon_Formulaire, Name => "ChampSel");
      Ma_Selection.Add_Option ("Valeur1", "Champ 1");
      Mon_Option1.Create (Mon_Formulaire, Ma_Selection, "Valeur2", "Champ 2");
      Mon_Option_Groupe.Create (Mon_Formulaire, Ma_Selection, "Groupe 1");
      Mon_Option2.Create (Mon_Formulaire, Mon_Option_Groupe, "Valeur3", "Champ 3");
    end Formulaires;

  begin
    Gnoga.Application.Title ("hello3");
    Gnoga.Application.HTML_On_Close ("<b>Connection to Application has been terminated</b>");

    --    Gnoga.Application.Open_URL ("http://127.0.0.1:8080");
    Gnoga.Application.Singleton.Initialize (Main_Window, Port => 8080);

    Main_View.Create (Main_Window);
    Elements_Basics;
    Main_View.Horizontal_Rule;
    Formulaires;

    Gnoga.Application.Singleton.Message_Loop;
  exception
    when E : others =>
      Gnoga.Log (Ada.Exceptions.Exception_Name (E) & " - " & Ada.Exceptions.Exception_Message
(E));
  end hello3;
```

Annexe E : le code complet de hello4.adb

```
with Ada.Exceptions;

with Gnoga.Application.Singleton;
with Gnoga.Gui.Window;
with Gnoga.Gui.View;
with Gnoga.Gui.View.Console;
with Gnoga.Gui.View.Grid;
with Gnoga.Gui.View.Docker;
with Gnoga.Gui.View.Card;
with Gnoga.Types.Colors;
with Gnoga.Gui.Element;
with Gnoga.Gui.Element.Common;
use Gnoga.Gui.View.Grid;
use Gnoga.Gui.Element;
use Gnoga.Types.Colors;

procedure hello4 is
  Main_Window       : Gnoga.Gui.Window.Window_Type;
  Main_View         : Gnoga.Gui.View.Grid.Grid_View_Type;
  Basic_View        : Gnoga.Gui.View.View_Type;
  Console_View      : Gnoga.Gui.View.Console.Console_View_Type;
  Grid_View       : Gnoga.Gui.View.Grid.Grid_View_Type;
  Docker_View       : Gnoga.Gui.View.Docker.Docker_View_Type;
  DL, DR, DT, DB, DF : aliased Gnoga.Gui.Element.Common.P_Type;
  Card_View         : Gnoga.Gui.View.Card.Card_View_Type;
  C1, C2            : aliased Gnoga.Gui.Element.Common.P_Type;
  Tabs              : Gnoga.Gui.View.Card.Tab_Type;

begin
  Gnoga.Application.Title ("hello4");
  Gnoga.Application.HTML_On_Close ("<b>Connection to Application has been terminated</b>");

  --    Gnoga.Application.Open_URL ("http://127.0.0.1:8080");
  Gnoga.Application.Singleton.Initialize (Main_Window, Port => 8080);

  Main_View.Create (Main_Window, ((1 => COL), (1 => COL), (1 => COL), (1 => COL), (1 =>
COL)));

  Basic_View.Create (Main_View.Panel (1, 1).all);
  Basic_View.Background_Color (To_String (Aqua));
  Basic_View.Put_Line ("Hello World! Basic");

  Console_View.Create (Main_View.Panel (2, 1).all);
  Console_View.Background_Color (To_String (Beige));
  Console_View.Put_Line ("Hello World! Console");

  Grid_View.Create (Main_View.Panel (3, 1).all, (1 => (COL, COL)), False);
  Grid_View.Background_Color (To_String (Cyan));
  Grid_View.Panel (1, 1).Background_Color (To_String (Fuchsia));
  Grid_View.Panel (1, 1).Put_Line ("Hello World! Grid 1, 1");
  Grid_View.Panel (1, 2).Background_Color (To_String (Forest_Green));
  Grid_View.Panel (1, 2).Put_Line ("Hello World! Grid 1, 2");

  Docker_View.Create (Main_View.Panel (4, 1).all);
```

```
      Docker_View.Fill_Parent;
      DL.Create (Docker_View, "Hello World! Dock left");
      DL.Background_Color (To_String (Light_Pink));
      Docker_View.Left_Dock (DL'Access);
      DR.Create (Docker_View, "Hello World! Dock right");
      DR.Background_Color (To_String (Light_Coral));
      Docker_View.Right_Dock (DR'Access);
      DT.Create (Docker_View, "Hello World! Dock top");
      DT.Background_Color (To_String (Light_Sea_Green));
      Docker_View.Top_Dock (DT'Access);
      DB.Create (Docker_View, "Hello World! Dock bottom");
      DB.Background_Color (To_String (Light_Yellow));
      Docker_View.Bottom_Dock (DB'Access);
      DF.Create (Docker_View, "Hello World! Dock fill");
      DF.Background_Color (To_String (Light_Gray));
      Docker_View.Fill_Dock (DF'Access);

      Card_View.Create (Main_View.Panel (5, 1).all);
      Card_View.Fill_Parent;
      Tabs.Create (Card_View, Card_View);
      C1.Create (Card_View, "Hello World! C1");
      C1.Background_Color (To_String (Sky_Blue));
      Card_View.Add_Card ("C1", C1'Access, True);
      Tabs.Add_Tab ("C1", "Card 1");
      C2.Create (Card_View, "Hello World! C2");
      C2.Background_Color (To_String (Royal_Blue));
      Card_View.Add_Card ("C2", C2'Access);
      Tabs.Add_Tab ("C2", "Card 2");

      Gnoga.Application.Singleton.Message_Loop;
   exception
      when E : others =>
         Gnoga.Log (Ada.Exceptions.Exception_Name (E) & " - " & Ada.Exceptions.Exception_Message
(E));
   end hello4;
```

Annexe F : le code complet de hello3get.adb, hello3post.adb et hello3evt.adb

```
with Ada.Exceptions;

with Gnoga.Gui.Window;
with Gnoga.Gui.View.Console;
with Gnoga.Gui.Element.Form;
with Gnoga.Types;
with Gnoga.Gui.Location;
with Gnoga.Application.Multi_Connect;

procedure hello3get is

  procedure Formulaires
   (Main_Window : in out Gnoga.Gui.Window.Window_Type'Class;
    Connection  :        access Gnoga.Application.Multi_Connect.Connection_Holder_Type)
  is
    pragma Unreferenced (Connection);
    Main_View           : Gnoga.Gui.View.Console.Console_View_Type;
    Mon_Formulaire      : Gnoga.Gui.Element.Form.Form_Type;
    Mon_Texte_Multi     : Gnoga.Gui.Element.Form.Text_Area_Type;
    Mon_Champ_Cache     : Gnoga.Gui.Element.Form.Hidden_Type;
    Mon_Bouton          : Gnoga.Gui.Element.Form.Input_Button_Type;
    Mon_Bouton_Envoi    : Gnoga.Gui.Element.Form.Submit_Button_Type;
    Mon_Bouton_RAZ      : Gnoga.Gui.Element.Form.Reset_Button_Type;
    Ma_Case_A_Cocher    : Gnoga.Gui.Element.Form.Check_Box_Type;
    Mon_Bouton_Radio1   : Gnoga.Gui.Element.Form.Radio_Button_Type;
    Mon_Bouton_Radio2   : Gnoga.Gui.Element.Form.Radio_Button_Type;
    Mon_Image           : Gnoga.Gui.Element.Form.Input_Image_Type;
    Mon_Texte           : Gnoga.Gui.Element.Form.Text_Type;
    Mon_Mel             : Gnoga.Gui.Element.Form.Email_Type;
    Mon_Mot_Passe       : Gnoga.Gui.Element.Form.Password_Type;
    Mon_URL             : Gnoga.Gui.Element.Form.URL_Type;
    Ma_Recherche        : Gnoga.Gui.Element.Form.Search_Type;
    Ma_Couleur          : Gnoga.Gui.Element.Form.Color_Picker_Type;
    Ma_Date             : Gnoga.Gui.Element.Form.Date_Type;
    Mon_Heure           : Gnoga.Gui.Element.Form.Time_Type;
    Mon_Mois            : Gnoga.Gui.Element.Form.Month_Type;
    Ma_Semaine          : Gnoga.Gui.Element.Form.Week_Type;
    Ma_Date_Heure_Locale : Gnoga.Gui.Element.Form.Date_Time_Local_Type;
    Mon_Nombre          : Gnoga.Gui.Element.Form.Number_Type;
    Mon_Glisseur        : Gnoga.Gui.Element.Form.Range_Type;
    Mon_Etiquette       : Gnoga.Gui.Element.Form.Label_Type;
    Ma_Selection        : Gnoga.Gui.Element.Form.Selection_Type;
    Mon_Option1         : Gnoga.Gui.Element.Form.Option_Type;
    Mon_Option2         : Gnoga.Gui.Element.Form.Option_Type;
    Mon_Option_Groupe   : Gnoga.Gui.Element.Form.Option_Group_Type;
  begin
    Main_View.Create (Main_Window);
    Mon_Formulaire.Create (Main_View, "/resultats");
    Mon_Texte_Multi.Create
    (Mon_Formulaire, Value => "Texte multi-ligne...", Name => "Texte multi-ligne");
    Mon_Formulaire.New_Line;
    Mon_Champ_Cache.Create (Mon_Formulaire, Value => "Valeur0", Name => "Champ Caché");
    Mon_Formulaire.Put ("<- Champ caché.");
    Mon_Formulaire.New_Line;
```

```
         Mon_Bouton.Create (Mon_Formulaire, "Annuler");
         Mon_Bouton_Envoi.Create (Mon_Formulaire, "Envoyer");
         Mon_Bouton_RAZ.Create (Mon_Formulaire, "RAZ");
         Mon_Formulaire.New_Line;
         Ma_Case_A_Cocher.Create (Mon_Formulaire, Value => "Cochée", Name => "Case à Cocher");
         Gnoga.Gui.Element.Form.Label_Access'(new Gnoga.Gui.Element.Form.Label_Type).Create
         (Mon_Formulaire, Ma_Case_A_Cocher, "Case à cocher");
         Mon_Formulaire.New_Line;
         Mon_Bouton_Radio1.Create (Mon_Formulaire, Value => "Choix1", Name => "Boutons radio");
         Gnoga.Gui.Element.Form.Label_Access'(new Gnoga.Gui.Element.Form.Label_Type).Create
         (Mon_Formulaire, Mon_Bouton_Radio1, "Choix 1");
         Mon_Bouton_Radio2.Create (Mon_Formulaire, True, Value => "Choix2", Name => "Boutons
radio");
         Gnoga.Gui.Element.Form.Label_Access'(new Gnoga.Gui.Element.Form.Label_Type).Create
         (Mon_Formulaire, Mon_Bouton_Radio2, "Choix 2");
         Mon_Formulaire.New_Line;
         Mon_Image.Create (Mon_Formulaire, "favicon.ico", Value => "favicon.ico", Name => "Image");
         Mon_Formulaire.Put ("<- image.");
         Mon_Formulaire.New_Line;
         Mon_Texte.Create (Mon_Formulaire, Name => "Texte");
         Mon_Formulaire.Put ("<- texte sur une ligne.");
         Mon_Formulaire.New_Line;
         Mon_Mel.Create (Mon_Formulaire, Value => "mel@moi.org", Name => "Mel");
         Mon_Formulaire.New_Line;
         Mon_Mot_Passe.Create (Mon_Formulaire, Value => "mdp", Name => "MDP");
         Mon_Formulaire.Put ("<- mot de passe.");
         Mon_Formulaire.New_Line;
         Mon_URL.Create (Mon_Formulaire, Value => "http://gnoga.com", Name => "URL");
         Mon_Formulaire.New_Line;
         Ma_Recherche.Create (Mon_Formulaire, Value => "gnoga", Name => "Recherche");
         Mon_Formulaire.Put ("<- recherche.");
         Mon_Formulaire.New_Line;
         Ma_Couleur.Create (Mon_Formulaire, Name => "Couleur");
         Mon_Formulaire.Put ("<- couleur.");
         Mon_Formulaire.New_Line;
         Ma_Date.Create (Mon_Formulaire, Name => "Date");
         Mon_Formulaire.Put ("<- date yyyy-mm-dd.");
         Mon_Formulaire.New_Line;
         Mon_Heure.Create (Mon_Formulaire, Name => "Heure");
         Mon_Formulaire.Put ("<- heure HH:MM.");
         Mon_Formulaire.New_Line;
         Mon_Mois.Create (Mon_Formulaire, Name => "Mois");
         Mon_Formulaire.Put ("<- mois yyyy-mm.");
         Mon_Formulaire.New_Line;
         Ma_Semaine.Create (Mon_Formulaire, Name => "Semaine");
         Mon_Formulaire.Put ("<- semaine yyyy-Www.");
         Mon_Formulaire.New_Line;
         Ma_Date_Heure_Locale.Create (Mon_Formulaire, Name => "Date heure locale");
         Mon_Formulaire.Put ("<- date et heure locale yyyy-mm-ddTHH:MMZ.");
         Mon_Formulaire.New_Line;
         Mon_Nombre.Create (Mon_Formulaire, Value => "99", Name => "Nombre");
         Mon_Formulaire.Put ("<- nombre.");
         Mon_Formulaire.New_Line;
         Mon_Glisseur.Create (Mon_Formulaire, Name => "Glisseur");
         Mon_Etiquette.Create (Mon_Formulaire, Mon_Glisseur, "Glisseur");
         Mon_Formulaire.New_Line;
```

```
      Ma_Selection.Create (Mon_Formulaire, Name => "Sélection");
      Ma_Selection.Add_Option ("Valeur1", "Champ 1");
      Mon_Option1.Create (Mon_Formulaire, Ma_Selection, "Valeur2", "Champ 2");
      Mon_Option_Groupe.Create (Mon_Formulaire, Ma_Selection, "Groupe 1");
      Mon_Option2.Create (Mon_Formulaire, Mon_Option_Groupe, "Valeur3", "Champ 3");
    end Formulaires;

    procedure Resultats
     (Main_Window : in out Gnoga.Gui.Window.Window_Type'Class;
      Connection  :        access Gnoga.Application.Multi_Connect.Connection_Holder_Type)
    is
      pragma Unreferenced (Connection);
      Main_View : Gnoga.Gui.View.Console.Console_View_Type;
    begin
      Main_View.Create (Main_Window);
      if Main_Window.Location.Search /= "" then
        Main_View.Put_Line (Main_Window.Document.Input_Encoding);
        Main_View.Put_Line ("Get Results: " & Main_Window.Location.Search);
        for C in
          Gnoga.Gui.Location.Parse
           (Main_Window.Location.Search,
            Main_Window.Document.Input_Encoding).Iterate
        loop
          begin
            Main_View.Put_Line
            ("GET parameter: " &
             Gnoga.Types.Data_Maps.Key (C) &
             " = " &
             Gnoga.Types.Data_Maps.Element (C));
          end;
        end loop;
      end if;
    end Resultats;

  begin
    Gnoga.Application.Title ("hello3 GET");
    Gnoga.Application.HTML_On_Close ("<b>Connection to Application has been terminated</b>");

    --     Gnoga.Application.Open_URL ("http://127.0.0.1:8080");
    Gnoga.Application.Multi_Connect.Initialize;
    Gnoga.Application.Multi_Connect.On_Connect_Handler (Formulaires'Unrestricted_Access);
    Gnoga.Application.Multi_Connect.On_Connect_Handler
     (Resultats'Unrestricted_Access,
      "/resultats");

    Gnoga.Application.Multi_Connect.Message_Loop;
  exception
    when E : others =>
      Gnoga.Log (Ada.Exceptions.Exception_Name (E) & " - " & Ada.Exceptions.Exception_Message
(E));
  end hello3get;
```

```
with Ada.Exceptions;
with Ada.Strings.Unbounded;

with Gnoga.Gui.Window;
with Gnoga.Gui.View.Console;
with Gnoga.Gui.Element.Form;
with Gnoga.Types;
with Gnoga.Application.Multi_Connect;
with Gnoga.Server.Connection;

procedure hello3post is

  Last_Parameters : Gnoga.Types.Data_Map_Type;

  procedure Formulaires
   (Main_Window : in out Gnoga.Gui.Window.Window_Type'Class;
    Connection  :        access Gnoga.Application.Multi_Connect.Connection_Holder_Type)
  is
    pragma Unreferenced (Connection);
    Main_View            : Gnoga.Gui.View.Console.Console_View_Type;
    Mon_Formulaire       : Gnoga.Gui.Element.Form.Form_Type;
    Mon_Texte_Multi      : Gnoga.Gui.Element.Form.Text_Area_Type;
    Mon_Champ_Cache      : Gnoga.Gui.Element.Form.Hidden_Type;
    Mon_Bouton           : Gnoga.Gui.Element.Form.Input_Button_Type;
    Mon_Bouton_Envoi     : Gnoga.Gui.Element.Form.Submit_Button_Type;
    Mon_Bouton_RAZ       : Gnoga.Gui.Element.Form.Reset_Button_Type;
    Ma_Case_A_Cocher     : Gnoga.Gui.Element.Form.Check_Box_Type;
    Mon_Bouton_Radio1    : Gnoga.Gui.Element.Form.Radio_Button_Type;
    Mon_Bouton_Radio2    : Gnoga.Gui.Element.Form.Radio_Button_Type;
    Mon_Image            : Gnoga.Gui.Element.Form.Input_Image_Type;
    Mon_Texte            : Gnoga.Gui.Element.Form.Text_Type;
    Mon_Mel              : Gnoga.Gui.Element.Form.Email_Type;
    Mon_Mot_Passe        : Gnoga.Gui.Element.Form.Password_Type;
    Mon_URL              : Gnoga.Gui.Element.Form.URL_Type;
    Ma_Recherche         : Gnoga.Gui.Element.Form.Search_Type;
    Ma_Couleur           : Gnoga.Gui.Element.Form.Color_Picker_Type;
    Ma_Date              : Gnoga.Gui.Element.Form.Date_Type;
    Mon_Heure            : Gnoga.Gui.Element.Form.Time_Type;
    Mon_Mois             : Gnoga.Gui.Element.Form.Month_Type;
    Ma_Semaine           : Gnoga.Gui.Element.Form.Week_Type;
    Ma_Date_Heure_Locale : Gnoga.Gui.Element.Form.Date_Time_Local_Type;
    Mon_Nombre           : Gnoga.Gui.Element.Form.Number_Type;
    Mon_Glisseur         : Gnoga.Gui.Element.Form.Range_Type;
    Mon_Etiquette        : Gnoga.Gui.Element.Form.Label_Type;
    Ma_Selection         : Gnoga.Gui.Element.Form.Selection_Type;
    Mon_Option1          : Gnoga.Gui.Element.Form.Option_Type;
    Mon_Option2          : Gnoga.Gui.Element.Form.Option_Type;
    Mon_Option_Groupe    : Gnoga.Gui.Element.Form.Option_Group_Type;
  begin
    Main_View.Create (Main_Window);
    Mon_Formulaire.Create (Main_View, "/resultats", Gnoga.Gui.Element.Form.Post);
    Mon_Texte_Multi.Create
    (Mon_Formulaire, Value => "Texte multi-ligne...", Name => "Texte multi-ligne");
    Mon_Formulaire.New_Line;
    Mon_Champ_Cache.Create (Mon_Formulaire, Value => "Valeur0", Name => "Champ Cache");
    Mon_Formulaire.Put ("<- Champ caché.");
```

```
         Mon_Formulaire.New_Line;
         Mon_Bouton.Create (Mon_Formulaire, "Annuler");
         Mon_Bouton_Envoi.Create (Mon_Formulaire, "Envoyer");
         Mon_Bouton_RAZ.Create (Mon_Formulaire, "RAZ");
         Mon_Formulaire.New_Line;
         Ma_Case_A_Cocher.Create (Mon_Formulaire, Value => "Cochée", Name => "Case a Cocher");
         Gnoga.Gui.Element.Form.Label_Access'(new Gnoga.Gui.Element.Form.Label_Type).Create
         (Mon_Formulaire, Ma_Case_A_Cocher, "Case à cocher");
         Mon_Formulaire.New_Line;
         Mon_Bouton_Radio1.Create (Mon_Formulaire, Value => "Choix1", Name => "Boutons radio");
         Gnoga.Gui.Element.Form.Label_Access'(new Gnoga.Gui.Element.Form.Label_Type).Create
         (Mon_Formulaire, Mon_Bouton_Radio1, "Choix 1");
         Mon_Bouton_Radio2.Create (Mon_Formulaire, True, Value => "Choix2", Name => "Boutons
radio");
         Gnoga.Gui.Element.Form.Label_Access'(new Gnoga.Gui.Element.Form.Label_Type).Create
         (Mon_Formulaire, Mon_Bouton_Radio2, "Choix 2");
         Mon_Formulaire.New_Line;
         Mon_Image.Create (Mon_Formulaire, "favicon.ico", Value => "favicon.ico", Name => "Image");
         Mon_Formulaire.Put ("<- image.");
         Mon_Formulaire.New_Line;
         Mon_Texte.Create (Mon_Formulaire, Name => "Texte");
         Mon_Formulaire.Put ("<- texte sur une ligne.");
         Mon_Formulaire.New_Line;
         Mon_Mel.Create (Mon_Formulaire, Value => "mel@moi.org", Name => "Mel");
         Mon_Formulaire.New_Line;
         Mon_Mot_Passe.Create (Mon_Formulaire, Value => "mdp", Name => "MDP");
         Mon_Formulaire.Put ("<- mot de passe.");
         Mon_Formulaire.New_Line;
         Mon_URL.Create (Mon_Formulaire, Value => "http://gnoga.com", Name => "URL");
         Mon_Formulaire.New_Line;
         Ma_Recherche.Create (Mon_Formulaire, Value => "gnoga", Name => "Recherche");
         Mon_Formulaire.Put ("<- recherche.");
         Mon_Formulaire.New_Line;
         Ma_Couleur.Create (Mon_Formulaire, Name => "Couleur");
         Mon_Formulaire.Put ("<- couleur.");
         Mon_Formulaire.New_Line;
         Ma_Date.Create (Mon_Formulaire, Name => "Date");
         Mon_Formulaire.Put ("<- date yyyy-mm-dd.");
         Mon_Formulaire.New_Line;
         Mon_Heure.Create (Mon_Formulaire, Name => "Heure");
         Mon_Formulaire.Put ("<- heure HH:MM.");
         Mon_Formulaire.New_Line;
         Mon_Mois.Create (Mon_Formulaire, Name => "Mois");
         Mon_Formulaire.Put ("<- mois yyyy-mm.");
         Mon_Formulaire.New_Line;
         Ma_Semaine.Create (Mon_Formulaire, Name => "Semaine");
         Mon_Formulaire.Put ("<- semaine yyyy-Www.");
         Mon_Formulaire.New_Line;
         Ma_Date_Heure_Locale.Create (Mon_Formulaire, Name => "Date heure locale");
         Mon_Formulaire.Put ("<- date et heure locale yyyy-mm-ddTHH:MMZ.");
         Mon_Formulaire.New_Line;
         Mon_Nombre.Create (Mon_Formulaire, Value => "99", Name => "Nombre");
         Mon_Formulaire.Put ("<- nombre.");
         Mon_Formulaire.New_Line;
         Mon_Glisseur.Create (Mon_Formulaire, Name => "Glisseur");
         Mon_Etiquette.Create (Mon_Formulaire, Mon_Glisseur, "Glisseur");
```

```
      Mon_Formulaire.New_Line;
      Ma_Selection.Create (Mon_Formulaire, Name => "Selection");
      Ma_Selection.Add_Option ("Valeur1", "Champ 1");
      Mon_Option1.Create (Mon_Formulaire, Ma_Selection, "Valeur2", "Champ 2");
      Mon_Option_Groupe.Create (Mon_Formulaire, Ma_Selection, "Groupe 1");
      Mon_Option2.Create (Mon_Formulaire, Mon_Option_Groupe, "Valeur3", "Champ 3");
    end Formulaires;

    procedure On_Post_Request
     (URI                : in     String;
      Accepted_Parameters :    out Ada.Strings.Unbounded.Unbounded_String)
    is
      pragma Unreferenced (URI);
    begin
      Accepted_Parameters :=
        Ada.Strings.Unbounded.To_Unbounded_String ("Texte multi-ligne,Champ Cache,Case a
Cocher,Boutons radio,Image,Texte,Mel,MDP,URL,Recherche,Couleur,Date,Heure,Mois,Semaine,Date
heure locale,Nombre,Glisseur,Selection,Image.x,Image.y");
    end On_Post_Request;

    procedure On_Post (URI : String; Parameters : in out Gnoga.Types.Data_Map_Type) is
      pragma Unreferenced (URI);
    begin
      Last_Parameters := Parameters;
    end On_Post;

    procedure Resultats
     (Main_Window : in out Gnoga.Gui.Window.Window_Type'Class;
      Connection  :        access Gnoga.Application.Multi_Connect.Connection_Holder_Type)
    is
      pragma Unreferenced (Connection);
      Main_View : Gnoga.Gui.View.Console.Console_View_Type;
    begin
      Main_View.Create (Main_Window);
      for C in Last_Parameters.Iterate loop
        begin
          Main_View.Put_Line
          ("POST parameter: " &
           Gnoga.Types.Data_Maps.Key (C) &
           " = " &
           Gnoga.Types.Data_Maps.Element (C));
        end;
      end loop;
      Last_Parameters.Clear;
    end Resultats;

  begin
    Gnoga.Application.Title ("hello3 POST");
    Gnoga.Application.HTML_On_Close ("<b>Connection to Application has been terminated</b>");

    --     Gnoga.Application.Open_URL ("http://127.0.0.1:8080");
    Gnoga.Application.Multi_Connect.Initialize;
    Gnoga.Application.Multi_Connect.On_Connect_Handler (Formulaires'Unrestricted_Access);
    Gnoga.Application.Multi_Connect.On_Connect_Handler
     (Resultats'Unrestricted_Access,
      "/resultats");
```

```
      Gnoga.Server.Connection.On_Post_Handler (On_Post'Unrestricted_Access);
      Gnoga.Server.Connection.On_Post_Request_Handler (On_Post_Request'Unrestricted_Access);

      Gnoga.Application.Multi_Connect.Message_Loop;
   exception
      when E : others =>
         Gnoga.Log (Ada.Exceptions.Exception_Name (E) & " - " & Ada.Exceptions.Exception_Message
(E));
   end hello3post;
```

```
with Ada.Exceptions;

with Gnoga.Application.Singleton;
with Gnoga.Gui.Window;
with Gnoga.Gui.View.Console;
with Gnoga.Gui.Element.Form;
with Gnoga.Types;
with Gnoga.Gui.Base;

procedure hello3evt is
  Main_Window : Gnoga.Gui.Window.Window_Type;

  type App_Data is new Gnoga.Types.Connection_Data_Type with record
    Main_View           : Gnoga.Gui.View.Console.Console_View_Type;
    Mon_Formulaire      : Gnoga.Gui.Element.Form.Form_Type;
    Mon_Texte_Multi     : Gnoga.Gui.Element.Form.Text_Area_Type;
    Mon_Champ_Cache     : Gnoga.Gui.Element.Form.Hidden_Type;
    Mon_Bouton          : Gnoga.Gui.Element.Form.Input_Button_Type;
    Mon_Bouton_Envoi    : Gnoga.Gui.Element.Form.Submit_Button_Type;
    Mon_Bouton_RAZ      : Gnoga.Gui.Element.Form.Reset_Button_Type;
    Ma_Case_A_Cocher    : Gnoga.Gui.Element.Form.Check_Box_Type;
    Mon_Bouton_Radio1   : Gnoga.Gui.Element.Form.Radio_Button_Type;
    Mon_Bouton_Radio2   : Gnoga.Gui.Element.Form.Radio_Button_Type;
    Mon_Image           : Gnoga.Gui.Element.Form.Input_Image_Type;
    Mon_Texte           : Gnoga.Gui.Element.Form.Text_Type;
    Mon_Mel             : Gnoga.Gui.Element.Form.Email_Type;
    Mon_Mot_Passe       : Gnoga.Gui.Element.Form.Password_Type;
    Mon_URL             : Gnoga.Gui.Element.Form.URL_Type;
    Ma_Recherche        : Gnoga.Gui.Element.Form.Search_Type;
    Ma_Couleur          : Gnoga.Gui.Element.Form.Color_Picker_Type;
    Ma_Date             : Gnoga.Gui.Element.Form.Date_Type;
    Mon_Heure           : Gnoga.Gui.Element.Form.Time_Type;
    Mon_Mois            : Gnoga.Gui.Element.Form.Month_Type;
    Ma_Semaine          : Gnoga.Gui.Element.Form.Week_Type;
    Ma_Date_Heure_Locale : Gnoga.Gui.Element.Form.Date_Time_Local_Type;
    Mon_Nombre          : Gnoga.Gui.Element.Form.Number_Type;
    Mon_Glisseur        : Gnoga.Gui.Element.Form.Range_Type;
    Mon_Etiquette       : Gnoga.Gui.Element.Form.Label_Type;
    Ma_Selection        : Gnoga.Gui.Element.Form.Selection_Type;
    Mon_Option1         : Gnoga.Gui.Element.Form.Option_Type;
    Mon_Option2         : Gnoga.Gui.Element.Form.Option_Type;
    Mon_Option_Groupe   : Gnoga.Gui.Element.Form.Option_Group_Type;
  end record;
  type App_Access is access all App_Data;

  procedure On_Cancel (Object : in out Gnoga.Gui.Base.Base_Type'Class) is
    App : constant App_Access := App_Access (Object.Connection_Data);
  begin
    App.Main_View.Put_HTML ("<h1>Annulation du formulaire</h1>");
  end On_Cancel;

  procedure On_Submit (Object : in out Gnoga.Gui.Base.Base_Type'Class) is
    App : constant App_Access := App_Access (Object.Connection_Data);
  begin
    App.Main_View.Put_Line ("EVT parameter: Mon_Texte_Multi = " & App.Mon_Texte_Multi.Value);
```

```
      App.Main_View.Put_Line ("EVT parameter: Mon_Champ_Cache = " &
App.Mon_Champ_Cache.Value);
      App.Main_View.Put_Line
      ("EVT parameter: Ma_Case_A_Cocher = " & App.Ma_Case_A_Cocher.Checked'Img);
      App.Main_View.Put_Line
      ("EVT parameter: Mon_Bouton_Radio1 = " & App.Mon_Bouton_Radio1.Checked'Img);
      App.Main_View.Put_Line
      ("EVT parameter: Mon_Bouton_Radio2 = " & App.Mon_Bouton_Radio2.Checked'Img);
      App.Main_View.Put_Line ("EVT parameter: Mon_Image = " & App.Mon_Image.Source);
      App.Main_View.Put_Line ("EVT parameter: Mon_Texte = " & App.Mon_Texte.Value);
      App.Main_View.Put_Line ("EVT parameter: Mon_Mel = " & App.Mon_Mel.Value);
      App.Main_View.Put_Line ("EVT parameter: Mon_Mot_Passe = " & App.Mon_Mot_Passe.Value);
      App.Main_View.Put_Line ("EVT parameter: Mon_URL = " & App.Mon_URL.Value);
      App.Main_View.Put_Line ("EVT parameter: Ma_Recherche = " & App.Ma_Recherche.Value);
      App.Main_View.Put_Line ("EVT parameter: Ma_Couleur = " & App.Ma_Couleur.Value);
      App.Main_View.Put_Line ("EVT parameter: Ma_Date = " & App.Ma_Date.Value);
      App.Main_View.Put_Line ("EVT parameter: Mon_Heure = " & App.Mon_Heure.Value);
      App.Main_View.Put_Line ("EVT parameter: Mon_Mois = " & App.Mon_Mois.Value);
      App.Main_View.Put_Line ("EVT parameter: Ma_Semaine = " & App.Ma_Semaine.Value);
      App.Main_View.Put_Line
      ("EVT parameter: Ma_Date_Heure_Locale = " & App.Ma_Date_Heure_Locale.Value);
      App.Main_View.Put_Line ("EVT parameter: Mon_Nombre = " & App.Mon_Nombre.Value);
      App.Main_View.Put_Line ("EVT parameter: Mon_Glisseur = " & App.Mon_Glisseur.Value);
      App.Main_View.Put_Line ("EVT parameter: Ma_Selection = " & App.Ma_Selection.Value);
    end On_Submit;

    procedure On_Change (Object : in out Gnoga.Gui.Base.Base_Type'Class) is
      App : constant App_Access := App_Access (Object.Connection_Data);
    begin
      App.Main_View.Put_Line ("EVT change: Mon_Glisseur = " & App.Mon_Glisseur.Value);
    end On_Change;

    procedure Formulaires (App : App_Access) is
    begin
      Main_Window.Connection_Data (App);
      App.Main_View.Create (Main_Window);
      App.Mon_Formulaire.Create (App.Main_View);
      App.Mon_Formulaire.On_Submit_Handler (On_Submit'Unrestricted_Access);
      App.Mon_Texte_Multi.Create (App.Mon_Formulaire, Value => "Texte multi-ligne...");
      App.Mon_Formulaire.New_Line;
      App.Mon_Champ_Cache.Create (App.Mon_Formulaire, Value => "Valeur0", Name =>
"Champ0");
      App.Mon_Formulaire.Put ("<- Champ caché.");
      App.Mon_Formulaire.New_Line;
      App.Mon_Bouton.Create (App.Mon_Formulaire, "Annuler");
      App.Mon_Bouton.On_Click_Handler (On_Cancel'Unrestricted_Access);
      App.Mon_Bouton_Envoi.Create (App.Mon_Formulaire, "Envoyer");
      App.Mon_Bouton_RAZ.Create (App.Mon_Formulaire, "RAZ");
      App.Mon_Formulaire.New_Line;
      App.Ma_Case_A_Cocher.Create (App.Mon_Formulaire);
      Gnoga.Gui.Element.Form.Label_Access'(new Gnoga.Gui.Element.Form.Label_Type).Create
      (App.Mon_Formulaire, App.Ma_Case_A_Cocher, "Case à cocher");
      App.Mon_Formulaire.New_Line;
      App.Mon_Bouton_Radio1.Create (App.Mon_Formulaire, Value => "Choix1", Name => "Choix");
      Gnoga.Gui.Element.Form.Label_Access'(new Gnoga.Gui.Element.Form.Label_Type).Create
      (App.Mon_Formulaire, App.Mon_Bouton_Radio1, "Choix 1");
```

```
         App.Mon_Bouton_Radio2.Create (App.Mon_Formulaire, True, Value => "Choix2", Name =>
"Choix");
         Gnoga.Gui.Element.Form.Label_Access'(new Gnoga.Gui.Element.Form.Label_Type).Create
         (App.Mon_Formulaire, App.Mon_Bouton_Radio2, "Choix 2");
         App.Mon_Formulaire.New_Line;
         App.Mon_Image.Create (App.Mon_Formulaire, "favicon.ico");
         App.Mon_Formulaire.Put ("<- image.");
         App.Mon_Formulaire.New_Line;
         App.Mon_Texte.Create (App.Mon_Formulaire);
         App.Mon_Formulaire.Put ("<- texte sur une ligne.");
         App.Mon_Formulaire.New_Line;
         App.Mon_Mel.Create (App.Mon_Formulaire, Value => "mel@moi.org");
         App.Mon_Formulaire.New_Line;
         App.Mon_Mot_Passe.Create (App.Mon_Formulaire, Value => "mdp");
         App.Mon_Formulaire.Put ("<- mot de passe.");
         App.Mon_Formulaire.New_Line;
         App.Mon_URL.Create (App.Mon_Formulaire, Value => "http://gnoga.com");
         App.Mon_Formulaire.New_Line;
         App.Ma_Recherche.Create (App.Mon_Formulaire, Value => "gnoga");
         App.Mon_Formulaire.Put ("<- recherche.");
         App.Mon_Formulaire.New_Line;
         App.Ma_Couleur.Create (App.Mon_Formulaire);
         App.Mon_Formulaire.Put ("<- couleur.");
         App.Mon_Formulaire.New_Line;
         App.Ma_Date.Create (App.Mon_Formulaire);
         App.Mon_Formulaire.Put ("<- date yyyy-mm-dd.");
         App.Mon_Formulaire.New_Line;
         App.Mon_Heure.Create (App.Mon_Formulaire);
         App.Mon_Formulaire.Put ("<- heure HH:MM.");
         App.Mon_Formulaire.New_Line;
         App.Mon_Mois.Create (App.Mon_Formulaire);
         App.Mon_Formulaire.Put ("<- mois yyyy-mm.");
         App.Mon_Formulaire.New_Line;
         App.Ma_Semaine.Create (App.Mon_Formulaire);
         App.Mon_Formulaire.Put ("<- semaine yyyy-Www.");
         App.Mon_Formulaire.New_Line;
         App.Ma_Date_Heure_Locale.Create (App.Mon_Formulaire);
         App.Mon_Formulaire.Put ("<- date et heure locale yyyy-mm-ddTHH:MMZ.");
         App.Mon_Formulaire.New_Line;
         App.Mon_Nombre.Create (App.Mon_Formulaire, Value => "99");
         App.Mon_Formulaire.Put ("<- nombre.");
         App.Mon_Formulaire.New_Line;
         App.Mon_Glisseur.Create (App.Mon_Formulaire);
         App.Mon_Glisseur.On_Change_Handler (On_Change'Unrestricted_Access);
         App.Mon_Etiquette.Create (App.Mon_Formulaire, App.Mon_Glisseur, "Glisseur");
         App.Mon_Formulaire.New_Line;
         App.Ma_Selection.Create (App.Mon_Formulaire, Name => "ChampSel");
         App.Ma_Selection.Add_Option ("Valeur1", "Champ 1");
         App.Mon_Option1.Create (App.Mon_Formulaire, App.Ma_Selection, "Valeur2", "Champ 2");
         App.Mon_Option_Groupe.Create (App.Mon_Formulaire, App.Ma_Selection, "Groupe 1");
         App.Mon_Option2.Create (App.Mon_Formulaire, App.Mon_Option_Groupe, "Valeur3", "Champ
3");
      end Formulaires;

   begin
      Gnoga.Application.Title ("hello3 EVT");
```

```
      Gnoga.Application.HTML_On_Close ("<b>Connection to Application has been terminated</b>");

      --     Gnoga.Application.Open_URL ("http://127.0.0.1:8080");
      Gnoga.Application.Singleton.Initialize (Main_Window, Port => 8080);

      Formulaires (new App_Data);

      Gnoga.Application.Singleton.Message_Loop;
   exception
      when E : others =>
         Gnoga.Log (Ada.Exceptions.Exception_Name (E) & " - " & Ada.Exceptions.Exception_Message
(E));
   end hello3evt;
```

Annexe G : le code complet de hello5_list.adb, hello5_iframe.adb et hello5_av.adb

```
with Ada.Exceptions;

with Gnoga.Application.Singleton;
with Gnoga.Gui.Window;
with Gnoga.Gui.View.Console;
with Gnoga.Gui.Element.List;

procedure hello5_list is
  Main_Window : Gnoga.Gui.Window.Window_Type;
  Main_View   : Gnoga.Gui.View.Console.Console_View_Type;

  Unordered_List, Unordered_Sublist1, Unordered_Sublist2  :
Gnoga.Gui.Element.List.Unordered_List_Type;
  ULItem1, ULItem2, ULItem3, ULItem11, ULItem21, ULItem22 :
Gnoga.Gui.Element.List.List_Item_Type;

  Ordered_List, Ordered_Sublist1, Ordered_Sublist2       :
Gnoga.Gui.Element.List.Ordered_List_Type;
  OLItem1, OLItem2, OLItem3, OLItem11, OLItem21, OLItem22 :
Gnoga.Gui.Element.List.List_Item_Type;

  Definition_List     : Gnoga.Gui.Element.List.Definition_List_Type;
  Term1, Term2, Term3 : Gnoga.Gui.Element.List.Term_Type;
  Desc1, Desc2, Desc3 : Gnoga.Gui.Element.List.Description_Type;
begin
  Gnoga.Application.Title ("hello5");
  Gnoga.Application.HTML_On_Close ("<b>Connection to Application has been terminated</b>");

  Gnoga.Application.Open_URL ("http://127.0.0.1:8080");
  Gnoga.Application.Singleton.Initialize (Main_Window, Port => 8080);

  Main_View.Create (Main_Window);
  Main_View.Put_Line ("Unordered list");

  Unordered_List.Create (Main_View);

  ULItem1.Create (Unordered_List, "I1");

  Unordered_Sublist1.Create (Unordered_List);
  ULItem11.Create (Unordered_Sublist1, "SI11");

  ULItem2.Create (Unordered_List, "I2");

  Unordered_Sublist2.Create (Unordered_List);
  Unordered_Sublist2.List_Kind (Gnoga.Gui.Element.List.Square);

  ULItem21.Create (Unordered_Sublist2, "SI21");
  ULItem22.Create (Unordered_Sublist2, "SI22");

  ULItem3.Create (Unordered_List, "I3");

  Main_View.Put_Line ("Ordered list");
```

```
      Ordered_List.Create (Main_View);
      Ordered_List.List_Kind (Gnoga.Gui.Element.List.Decimal_Leading_Zero);

      OLItem1.Create (Ordered_List, "I1");
      OLItem2.Create (Ordered_List, "I2");
      OLItem3.Create (Ordered_List, "I3");

      Ordered_Sublist1.Create (Ordered_List);
      Ordered_Sublist1.Place_Inside_Bottom_Of (OLItem1);
      Ordered_Sublist2.Create (Ordered_List);
      Ordered_Sublist2.Place_Inside_Bottom_Of (OLItem2);
      Ordered_Sublist2.List_Location (Gnoga.Gui.Element.List.Inside);

      OLItem11.Create (Ordered_Sublist1, "SI11");
      OLItem21.Create (Ordered_Sublist2, "SI21");
      OLItem22.Create (Ordered_Sublist2, "SI22");

      Main_View.Put_Line ("Description list");

      Definition_List.Create (Main_View);

      Term1.Create (Definition_List, "I1");
      Desc1.Create (Definition_List, "-> D1");
      Term2.Create (Definition_List, "I2");
      Desc2.Create (Definition_List, "-> D2");
      Term3.Create (Definition_List, "I3");
      Desc3.Create (Definition_List, "-> D3");

      Gnoga.Log (ULItem21.Value);
      Gnoga.Log (OLItem21.Value);

      Gnoga.Application.Singleton.Message_Loop;
   exception
      when E : others =>
         Gnoga.Log (Ada.Exceptions.Exception_Name (E) & " - " & Ada.Exceptions.Exception_Message
(E));
   end hello5_list;
```

```
with Ada.Exceptions;

with Gnoga.Application.Singleton;
with Gnoga.Gui.Window;
with Gnoga.Gui.View.Console;
with Gnoga.Gui.Element.IFrame;

procedure hello5_iframe is
  Main_Window : Gnoga.Gui.Window.Window_Type;
  Main_View   : Gnoga.Gui.View.Console.Console_View_Type;

  Web_Frame : Gnoga.Gui.Element.IFrame.IFrame_Type;

begin
  Gnoga.Application.Title ("hello5");
  Gnoga.Application.HTML_On_Close ("<b>Connection to Application has been terminated</b>");

  Gnoga.Application.Open_URL ("http://127.0.0.1:8080");
  Gnoga.Application.Singleton.Initialize (Main_Window, Port => 8080);

  Main_View.Create (Main_Window);
  Main_View.Put_Line ("Gnoga web page (Internet connexion required):");

  Web_Frame.Create (Main_View, "http://www.gnoga.com");
  Web_Frame.Width (300);
  Web_Frame.Height (200);
  Web_Frame.Border;

  Gnoga.Application.Singleton.Message_Loop;
exception
  when E : others =>
    Gnoga.Log (Ada.Exceptions.Exception_Name (E) & " - " & Ada.Exceptions.Exception_Message
(E));
end hello5_iframe;
```

```
with Ada.Exceptions;

with Gnoga.Application.Singleton;
with Gnoga.Gui.Window;
with Gnoga.Gui.View.Console;
with Gnoga.Gui.Element.Multimedia;

procedure hello5_av is
  Main_Window : Gnoga.Gui.Window.Window_Type;
  Main_View   : Gnoga.Gui.View.Console.Console_View_Type;

  Audio : Gnoga.Gui.Element.Multimedia.Audio_Type;
  Video : Gnoga.Gui.Element.Multimedia.Video_Type;

begin
  Gnoga.Application.Title ("hello5");
  Gnoga.Application.HTML_On_Close ("<b>Connection to Application has been terminated</b>");

  Gnoga.Application.Open_URL ("http://127.0.0.1:8080");
  Gnoga.Application.Singleton.Initialize (Main_Window, Port => 8080);

  Main_View.Create (Main_Window);
  Main_View.Put_Line ("Audio");
  Audio.Create (Main_View, "img/tune.mp3");
  Gnoga.Log ("The browser can play MP3s? - " & Audio.Can_Play ("audio/mp3")'Img);
  Main_View.Put_Line ("Video");
  Video.Create (Main_View, "img/test.mp4");
  Gnoga.Log ("The browser can play MP4s? - " & Video.Can_Play ("video/mp4")'Img);

  Gnoga.Application.Singleton.Message_Loop;
exception
  when E : others =>
    Gnoga.Log (Ada.Exceptions.Exception_Name (E) & " - " & Ada.Exceptions.Exception_Message
(E));
end hello5_av;
```

Annexe H : le code complet de hello6

```
with Ada.Exceptions;

with Gnoga.Application.Multi_Connect;

with hello6.Controller;

procedure hello6.Main is
begin
  Gnoga.Application.Title ("hello6");
  Gnoga.Application.HTML_On_Close
   ("<b>Connection to Application has been terminated</b>");

  Gnoga.Application.Multi_Connect.Initialize;

  Gnoga.Application.Multi_Connect.Message_Loop;
exception
  when E : others =>
    Gnoga.Log (Ada.Exceptions.Exception_Name (E) & " - " &
                 Ada.Exceptions.Exception_Message (E));
end hello6.Main;

with Gnoga.Gui.Base;
with Gnoga.Gui.View;
with Gnoga.Gui.Element.Common;

package hello6.View is

  type Default_View_Type is new Gnoga.Gui.View.View_Type with
    record
      Label_Text   : Gnoga.Gui.View.View_Type;
      Click_Button : Gnoga.Gui.Element.Common.Button_Type;
    end record;
  type Default_View_Access is access all Default_View_Type;
  type Pointer_to_Default_View_Class is access all Default_View_Type'Class;

  overriding
  procedure Create
   (View   : in out Default_View_Type;
    Parent : in out Gnoga.Gui.Base.Base_Type'Class;
    ID     : in     String  := "");

end hello6.View;

package body hello6.View is

  ------------
  -- Create --
  ------------

  overriding
  procedure Create
   (View   : in out Default_View_Type;
    Parent : in out Gnoga.Gui.Base.Base_Type'Class;
    ID     : in     String  := "")
```

```
   is
   begin
      Gnoga.Gui.View.View_Type (View).Create (Parent, ID);

      View.Label_Text.Create (View);
      View.Click_Button.Create (View, "Click Me");
   end Create;

end hello6.View;

with Gnoga.Gui.Window;
with Gnoga.Application.Multi_Connect;
package hello6.Controller is
   pragma Elaborate_Body;
--     procedure Default
--       (Main_Window : in out Gnoga.Gui.Window.Window_Type'Class;
--        Connection  : access
--          Gnoga.Application.Multi_Connect.Connection_Holder_Type);
end hello6.Controller;

with Gnoga.Gui.Base;
with Gnoga.Gui.Element.Common;
with Gnoga.Types;
with Gnoga.Gui.View;

with hello6.View;
package body hello6.Controller is

   type App_Data_Type is new Gnoga.Types.Connection_Data_Type with record
      View : hello6.View.Default_View_Type;
   end record;
   type App_Data_Access is access all App_Data_Type;

   procedure On_Click1 (Object : in out Gnoga.Gui.Base.Base_Type'Class);

   procedure On_Click1 (Object : in out Gnoga.Gui.Base.Base_Type'Class) is
      View : hello6.View.Default_View_Access :=
               hello6.View.Default_View_Access (Object.Parent);
   begin
      View.Label_Text.Put_Line ("Click1");
   end On_Click1;

   procedure On_Click2 (Object : in out Gnoga.Gui.Base.Base_Type'Class);

   procedure On_Click2 (Object : in out Gnoga.Gui.Base.Base_Type'Class) is
   begin
      App_Data_Access(Object.Connection_Data).View.Label_Text.Put_Line ("Click2");
   end On_Click2;

   procedure On_Click3 (Object : in out Gnoga.Gui.Base.Base_Type'Class);

   procedure On_Click3 (Object : in out Gnoga.Gui.Base.Base_Type'Class) is
      Window : Gnoga.Gui.Window.Window_Access :=
        Gnoga.Gui.Window.Window_Access (Object.Parent.Parent);
      View : Gnoga.Gui.View.View_Type;
   begin
```

```
      Gnoga.Gui.View.View_Access(Window.Get_View).Remove;
      View.Create (Window.all);
      View.Put_Line ("Application exited.");
      Window.Close_Connection;
    end On_Click3;

    procedure Default1
      (Main_Window : in out Gnoga.Gui.Window.Window_Type'Class;
       Connection  : access
         Gnoga.Application.Multi_Connect.Connection_Holder_Type)
    is
      View : hello6.View.Default_View_Type;
    begin
      View.Create (Main_Window);
      View.Click_Button.On_Click_Handler (On_Click1'Access);
     Connection.Hold;
    end Default1;

    procedure Default2
      (Main_Window : in out Gnoga.Gui.Window.Window_Type'Class;
       Connection  : access
         Gnoga.Application.Multi_Connect.Connection_Holder_Type)
    is
      App_Data : App_Data_Access := new App_Data_Type;
    begin
      Main_Window.Connection_Data (App_Data);
      App_Data.View.Create (Main_Window);
      App_Data.View.Click_Button.On_Click_Handler (On_Click2'Access);
    end Default2;

    procedure Default3
      (Main_Window : in out Gnoga.Gui.Window.Window_Type'Class;
       Connection  : access
         Gnoga.Application.Multi_Connect.Connection_Holder_Type)
    is
      View : hello6.View.Default_View_Access :=
        new hello6.View.Default_View_Type;
      Close_Button : Gnoga.Gui.Element.Common.Button_Access :=
        new Gnoga.Gui.Element.Common.Button_Type;
    begin
      View.Dynamic;
      View.Create (Main_Window);
      View.Click_Button.On_Click_Handler (On_Click1'Access);
      Close_Button.Dynamic;
      Close_Button.Create (View.all, "Close");
      Close_Button.On_Click_Handler (On_Click3'Access);
    end Default3;

  begin
    Gnoga.Application.Multi_Connect.On_Connect_Handler
      (Default1'Access, "default1");
    Gnoga.Application.Multi_Connect.On_Connect_Handler
      (Default2'Access, "default2");
    Gnoga.Application.Multi_Connect.On_Connect_Handler
      (Default3'Access, "default3");
  end hello6.Controller;
```

Annexe I : le code complet de hello7_session

```
with Ada.Exceptions;
with Ada.Numerics.Discrete_Random;
with Ada.Containers.Indefinite_Ordered_Maps;

with Gnoga.Types;
with Gnoga.Gui.Window;
with Gnoga.Gui.Base;
with Gnoga.Gui.View.Console;
with Gnoga.Gui.Element.Common;
with Gnoga.Application.Multi_Connect;
with Gnoga.Client.Storage;

procedure hello7_session is

   subtype Char is Character range 'a' .. 'z';
   package Char_Random is new Ada.Numerics.Discrete_Random (Char);
   Char_Generator : Char_Random.Generator;

   package ID_Map is new Ada.Containers.Indefinite_Ordered_Maps (String, Character);
   Text_Map : ID_Map.Map;

   type App_Data_Type is new Gnoga.Types.Connection_Data_Type with record
      Main_View  : Gnoga.Gui.View.Console.Console_View_Type;
      Mon_Bouton : Gnoga.Gui.Element.Common.Button_Type;
      Session    : Gnoga.Client.Storage.Session_Storage_Type;
   end record;
   type App_Data_Access is access all App_Data_Type;

   procedure On_Inc (Object : in out Gnoga.Gui.Base.Base_Type'Class) is
      Client_Text : constant String :=
        App_Data_Access (Object.Connection_Data).Session.Get ("TEXT") &
        Text_Map.Element (App_Data_Access (Object.Connection_Data).Session.Get ("gid"));
   begin
      App_Data_Access (Object.Connection_Data).Session.Set ("TEXT", Client_Text);
      App_Data_Access (Object.Connection_Data).Main_View.Put_Line
      ("Texte serveur : " & Text_Map.Element (App_Data_Access
(Object.Connection_Data).Session.Get ("gid")));
      App_Data_Access (Object.Connection_Data).Main_View.Put_Line ("Texte client : " &
Client_Text);
   end On_Inc;

   procedure On_Connect
     (Main_Window : in out Gnoga.Gui.Window.Window_Type'Class;
      Connection  :        access Gnoga.Application.Multi_Connect.Connection_Holder_Type)
   is
      pragma Unreferenced (Connection);
      App_Data : constant App_Data_Access := new App_Data_Type;
   begin
      Main_Window.Connection_Data (App_Data);
      App_Data.Main_View.Create (Main_Window);
      App_Data.Mon_Bouton.Create (App_Data.Main_View, "Incrémenter du texte");
      App_Data.Mon_Bouton.On_Click_Handler (On_Inc'Unrestricted_Access);
      App_Data.Main_View.Put_Line ("gid de session : " & Main_Window.Gnoga_Session_ID);
      App_Data.Session := Gnoga.Client.Storage.Session_Storage (Main_Window);
```

```
            App_Data.Session.Set ("TEXT", " -> ");
            Char_Random.Reset (Char_Generator);
            Text_Map.Include (Main_Window.Gnoga_Session_ID, Char_Random.Random
(Char_Generator));
            App_Data.Main_View.Put_Line ("Texte serveur : " & Text_Map.Element
(Main_Window.Gnoga_Session_ID));
            App_Data.Main_View.Put_Line ("Texte client : " & App_Data.Session.Get ("TEXT"));
          end On_Connect;

       begin
          Gnoga.Application.Title ("hello7 session");
          Gnoga.Application.HTML_On_Close ("<b>Connection to Application has been terminated</b>");

          --     Gnoga.Application.Open_URL ("http://127.0.0.1:8080");
          Gnoga.Application.Multi_Connect.Initialize;
          Gnoga.Application.Multi_Connect.On_Connect_Handler (On_Connect'Unrestricted_Access);

          Gnoga.Application.Multi_Connect.Message_Loop;
       exception
          when E : others =>
            Gnoga.Log (Ada.Exceptions.Exception_Name (E) & " - " & Ada.Exceptions.Exception_Message
(E));
       end hello7_session;
```

Fin de l'article.