2.5 INFORMACIÓN GENERAL

Dlg_InfGral.h [Diseño]

El primer formulario secundario, el Diálogo de Información General, tiene esta interfaz compacta en la que se aportan todos los datos necesarios para concretar la lectura de un otolito.

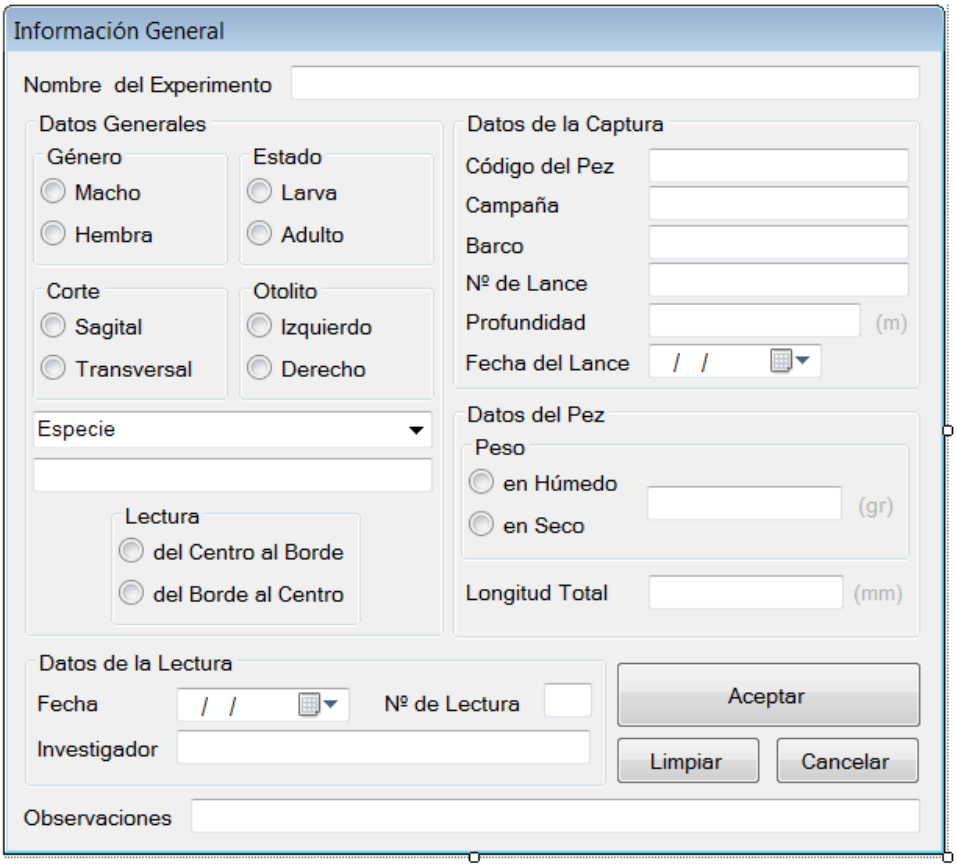

Figura 24.- Formulario de Información General

Dlg_InfGral.h

Este es el archivo con más líneas de código de toda la aplicación, y no se debe a su complejidad, si no a la cantidad de conexiones que hay que realizar para que sea a prueba de fallos. En este capítulo abordaremos la cuestión de cómo hacer un formulario robusto, al que no se le puedan introducir datos inválidos que acarrearían errores posteriores, y normalmente difíciles de localizar. Estudiaremos su código detenidamente.

```
1     #pragma once
2
...   ...
9
10    // para acortar codigo
11    #define DlgRes System::Windows::Forms::DialogResult
12
13    namespace OTOLIVE {
14
15       /// <summary>
16       /// Resumen de Dlg_InfGral
17       ///
18       /// ADVERTENCIA: si cambia el nombre de esta clase, deberá cambiar la
19       ///          propiedad 'Nombre de archivos de recursos' de la herramienta de
      compilación de recursos administrados
20       ///          asociada con todos los archivos .resx de los que depende esta
      clase. De lo contrario,
21       ///          los diseñadores no podrán interactuar correctamente con los
22       ///          recursos adaptados asociados con este formulario.
23       /// </summary>
24       public ref class Dlg_InfGral : public System::Windows::Forms::Form
25       {
26       public:
27          // definicion de variables para comunicacion con otros procesos, internos y
      externos
28          Dlg_InfGral(void) :
29                condNombreExp(false)
30             , condMacho(false)
31             , condHembra(false)
32             , condLarva(false)
33             , condAdulto(false)
34             , condSagital(false)
35             , condTransversal(false)
36             , condIzquierdo(false)
37             , condDerecho(false)
38             , condEspecie(false)
39             , condLecCB(false)
40             , condLecBC(false)
41             , condCodigo(false)
42             , condCampana(false)
43             , condBarco(false)
44             , condNumLance(false)
45             , condProfundidad(false)
46             , condFechaLance(false)
47             , condPesoH(false)
48             , condPesoS(false)
49             , condPeso(false)
50             , condLongitud(false)
51             , condFechaLec(false)
52             , condNumLec(false)
53             , condNombreInv(false)
54             , condObservaciones(false)
55          {
56             InitializeComponent();
57             //
58             //TODO: agregar código de constructor aquí
59             //
60          }
61
62       protected:
63          /// <summary>
64          /// Limpiar los recursos que se estén utilizando.
65          /// </summary>
66          ~Dlg_InfGral()
67          {
68             if (components)
69             {
70                delete components;
71             }
72          }
73
...   ...
908   #pragma endregion
909
910         // variables de comprobacion de errores
911         bool condNombreExp;
912         bool condMacho;
```

```
913         bool condHembra;
914         bool condLarva;
915         bool condAdulto;
916         bool condSagital;
917         bool condTransversal;
918         bool condIzquierdo;
919         bool condDerecho;
920         bool condEspecie;
921         bool condLecCB;
922         bool condLecBC;
923         bool condCodigo;
924         bool condCampana;
925         bool condBarco;
926         bool condNumLance;
927         bool condProfundidad;
928         bool condFechaLance;
929         bool condPesoH;
930         bool condPesoS;
931         bool condPeso;
932         bool condLongitud;
933         bool condFechaLec;
934         bool condNumLec;
935         bool condNombreInv;
936         bool condObservaciones;
937         // para usar #define DlgRes System::Windows::Forms::DialogResult y acortar
    codigo
938         DlgRes result;
939
940         // propiedades para el envio de datos
941         public: property String^ NombreExp
942         {
943            String^ get()
944            {
945               return textBoxNombreExp->Text;
946            }
947         }
948
949         public: property String^ Genero
950         {
951            String^ get()
952            {
953               if(rdBtnGenM->Checked)
954                  return "Macho";
955               else if (rdBtnGenH->Checked)
956                  return "Hembra";
957               else
958                  return "";
959            }
960         }
961
962         public: property String^ Estado
963         {
964            String^ get()
965            {
966               if(rdBtnEstL->Checked)
967                  return "Larva";
968               else if (rdBtnEstA->Checked)
969                  return "Adulto";
970               else
971                  return "";
972            }
973         }
974
975         public: property String^ Corte
976         {
977            String^ get()
978            {
979               if(rdBtnCorS->Checked)
980                  return "Sagital";
981               else if(rdBtnCorT->Checked)
982                  return "Transversal";
983               else
984                  return "";
985            }
986         }
987
988         public: property String^ Otolito
```

```
989         {
990            String^ get()
991            {
992               if(rdBtnOtoI->Checked)
993                  return "Izquierdo";
994               else if(rdBtnOtoD->Checked)
995                  return "Derecho";
996               else
997                  return "";
998            }
999         }
1000
1001        public: property String^ Especie
1002        {
1003           String^ get()
1004           {
1005              return textBoxEspecie->Text;
1006           }
1007        }
1008
1009        public: property String^ Lectura
1010        {
1011           String^ get()
1012           {
1013              if(rdBtnLecCB->Checked)
1014                 return "del Centro al Borde";
1015              else if(rdBtnLecBC->Checked)
1016                 return "del Borde al Centro";
1017              else
1018                 return "";
1019           }
1020        }
1021
1022        public: property String^ Codigo
1023        {
1024           String^ get()
1025           {
1026              return textBoxCodigo->Text;
1027           }
1028        }
1029
1030        public: property String^ Campana
1031        {
1032           String^ get()
1033           {
1034              return textBoxCampana->Text;
1035           }
1036        }
1037
1038        public: property String^ Barco
1039        {
1040           String^ get()
1041           {
1042              return textBoxBarco->Text;
1043           }
1044        }
1045
1046        public: property String^ NumLance
1047        {
1048           String^ get()
1049           {
1050              return textBoxNumLance->Text;
1051           }
1052        }
1053
1054        public: property String^ Profundidad
1055        {
1056           String^ get()
1057           {
1058              return textBoxProfundidad->Text;
1059           }
1060        }
1061
1062        public: property String^ FechaLance
1063        {
1064           String^ get()
1065           {
```

```
1066            return dateTimePicker1->Text;
1067         }
1068      }
1069
1070      public: property String^ RdPeso
1071      {
1072         String^ get()
1073         {
1074            if(rdBtnPesoH->Checked)
1075               return "en Húmedo";
1076            else if(rdBtnPesoS->Checked)
1077               return "en Seco";
1078            else
1079               return "";
1080         }
1081      }
1082
1083      public: property String^ Peso
1084      {
1085         String^ get()
1086         {
1087            return textBoxPeso->Text;
1088         }
1089      }
1090
1091      public: property String^ Longitud
1092      {
1093         String^ get()
1094         {
1095            return textBoxLongitud->Text;
1096         }
1097      }
1098
1099      public: property String^ FechaLec
1100      {
1101         String^ get()
1102         {
1103            return dateTimePicker2->Text;
1104         }
1105      }
1106
1107      public: property String^ NumLec
1108      {
1109         String^ get()
1110         {
1111            return textBoxNumLec->Text;
1112         }
1113      }
1114
1115      public: property String^ NombreInv
1116      {
1117         String^ get()
1118         {
1119            return textBoxNombreInv->Text;
1120         }
1121      }
1122
1123      public: property String^ Observaciones
1124      {
1125         String^ get()
1126         {
1127            return textBoxObservaciones->Text;
1128         }
1129      }
1130
1131      // comprobaciones al rellenar los campos
1132      private: System::Void textBoxNombreExp_TextChanged(System::Object^  sender,
System::EventArgs^  e)
1133      {
1134         // validamos la condicion
1135         condNombreExp = true;
1136         // comprobamos si es la ultima validada para devolver a Form_Ppal el
valor correcto
1137         if (condNombreExp && (condMacho || condHembra) && (condLarva ||
condAdulto) && (condSagital || condTransversal) && (condIzquierdo || condDerecho)
&& condEspecie && (condLecCB || condLecBC) && condCodigo && condCampana &&
condBarco && condNumLance && condProfundidad && condFechaLance && (condPesoH ||
```

```
      condPesoS) && condPeso && condLongitud && condFechaLec && condNumLec &&
      condNombreInv)
1138              btnOK_InfGral->DialogResult::set(DlgRes::OK);
1139           // ya se puede limpiar
1140           btnLimpiar_InfGral->Enabled = true;
1141           btnOK_InfGral->Enabled = true;
1142        }
1143
1144        private: System::Void rdBtnGenM_CheckedChanged(System::Object^  sender,
      System::EventArgs^  e)
1145        {
1146           condMacho = true;
1147           condHembra = false;
1148           if (condNombreExp && (condMacho || condHembra) && (condLarva ||
      condAdulto) && (condSagital || condTransversal) && (condIzquierdo || condDerecho)
      && condEspecie && (condLecCB || condLecBC) && condCodigo && condCampana &&
      condBarco && condNumLance && condProfundidad && condFechaLance && (condPesoH ||
      condPesoS) && condPeso && condLongitud && condFechaLec && condNumLec &&
      condNombreInv)
1149              btnOK_InfGral->DialogResult::set(DlgRes::OK);
1150           btnLimpiar_InfGral->Enabled = true;
1151           btnOK_InfGral->Enabled = true;
1152        }
1153
1154        private: System::Void rdBtnGenH_CheckedChanged(System::Object^  sender,
      System::EventArgs^  e)
1155        {
1156           condHembra = true;
1157           condMacho = false;
1158           if (condNombreExp && (condMacho || condHembra) && (condLarva ||
      condAdulto) && (condSagital || condTransversal) && (condIzquierdo || condDerecho)
      && condEspecie && (condLecCB || condLecBC) && condCodigo && condCampana &&
      condBarco && condNumLance && condProfundidad && condFechaLance && (condPesoH ||
      condPesoS) && condPeso && condLongitud && condFechaLec && condNumLec &&
      condNombreInv)
1159              btnOK_InfGral->DialogResult::set(DlgRes::OK);
1160           btnLimpiar_InfGral->Enabled = true;
1161           btnOK_InfGral->Enabled = true;
1162        }
1163
1164        private: System::Void rdBtnEstL_CheckedChanged(System::Object^  sender,
      System::EventArgs^  e)
1165        {
1166           condLarva = true;
1167           condAdulto = false;
1168           if (condNombreExp && (condMacho || condHembra) && (condLarva ||
      condAdulto) && (condSagital || condTransversal) && (condIzquierdo || condDerecho)
      && condEspecie && (condLecCB || condLecBC) && condCodigo && condCampana &&
      condBarco && condNumLance && condProfundidad && condFechaLance && (condPesoH ||
      condPesoS) && condPeso && condLongitud && condFechaLec && condNumLec &&
      condNombreInv)
1169              btnOK_InfGral->DialogResult::set(DlgRes::OK);
1170           btnLimpiar_InfGral->Enabled = true;
1171           btnOK_InfGral->Enabled = true;
1172        }
1173
1174        private: System::Void rdBtnEstA_CheckedChanged(System::Object^  sender,
      System::EventArgs^  e)
1175        {
1176           condAdulto = true;
1177           condLarva = false;
1178           if (condNombreExp && (condMacho || condHembra) && (condLarva ||
      condAdulto) && (condSagital || condTransversal) && (condIzquierdo || condDerecho)
      && condEspecie && (condLecCB || condLecBC) && condCodigo && condCampana &&
      condBarco && condNumLance && condProfundidad && condFechaLance && (condPesoH ||
      condPesoS) && condPeso && condLongitud && condFechaLec && condNumLec &&
      condNombreInv)
1179              btnOK_InfGral->DialogResult::set(DlgRes::OK);
1180           btnLimpiar_InfGral->Enabled = true;
1181           btnOK_InfGral->Enabled = true;
1182        }
1183
1184        private: System::Void rdBtnCorS_CheckedChanged(System::Object^  sender,
      System::EventArgs^  e)
1185        {
1186           condSagital = true;
1187           condTransversal = false;
```

```
1188            if (condNombreExp && (condMacho || condHembra) && (condLarva ||
    condAdulto) && (condSagital || condTransversal) && (condIzquierdo || condDerecho)
    && condEspecie && (condLecCB || condLecBC) && condCodigo && condCampana &&
    condBarco && condNumLance && condProfundidad && condFechaLance && (condPesoH ||
    condPesoS) && condPeso && condLongitud && condFechaLec && condNumLec &&
    condNombreInv)
1189                btnOK_InfGral->DialogResult::set(DlgRes::OK);
1190            btnLimpiar_InfGral->Enabled = true;
1191            btnOK_InfGral->Enabled = true;
1192        }
1193
1194        private: System::Void rdBtnCorT_CheckedChanged(System::Object^  sender,
    System::EventArgs^  e)
1195        {
1196            condTransversal = true;
1197            condSagital = false;
1198            if (condNombreExp && (condMacho || condHembra) && (condLarva ||
    condAdulto) && (condSagital || condTransversal) && (condIzquierdo || condDerecho)
    && condEspecie && (condLecCB || condLecBC) && condCodigo && condCampana &&
    condBarco && condNumLance && condProfundidad && condFechaLance && (condPesoH ||
    condPesoS) && condPeso && condLongitud && condFechaLec && condNumLec &&
    condNombreInv)
1199                btnOK_InfGral->DialogResult::set(DlgRes::OK);
1200            btnLimpiar_InfGral->Enabled = true;
1201            btnOK_InfGral->Enabled = true;
1202        }
1203
1204        private: System::Void rdBtnOtoI_CheckedChanged(System::Object^  sender,
    System::EventArgs^  e)
1205        {
1206            condIzquierdo = true;
1207            condDerecho = false;
1208            if (condNombreExp && (condMacho || condHembra) && (condLarva ||
    condAdulto) && (condSagital || condTransversal) && (condIzquierdo || condDerecho)
    && condEspecie && (condLecCB || condLecBC) && condCodigo && condCampana &&
    condBarco && condNumLance && condProfundidad && condFechaLance && (condPesoH ||
    condPesoS) && condPeso && condLongitud && condFechaLec && condNumLec &&
    condNombreInv)
1209                btnOK_InfGral->DialogResult::set(DlgRes::OK);
1210            btnLimpiar_InfGral->Enabled = true;
1211            btnOK_InfGral->Enabled = true;
1212        }
1213
1214        private: System::Void rdBtnOtoD_CheckedChanged(System::Object^  sender,
    System::EventArgs^  e)
1215        {
1216            condDerecho = true;
1217            condIzquierdo = false;
1218            if (condNombreExp && (condMacho || condHembra) && (condLarva ||
    condAdulto) && (condSagital || condTransversal) && (condIzquierdo || condDerecho)
    && condEspecie && (condLecCB || condLecBC) && condCodigo && condCampana &&
    condBarco && condNumLance && condProfundidad && condFechaLance && (condPesoH ||
    condPesoS) && condPeso && condLongitud && condFechaLec && condNumLec &&
    condNombreInv)
1219                btnOK_InfGral->DialogResult::set(DlgRes::OK);
1220            btnLimpiar_InfGral->Enabled = true;
1221            btnOK_InfGral->Enabled = true;
1222        }
1223
1224        private: System::Void comboBoxEspecie_KeyPress(System::Object^  sender,
    System::Windows::Forms::KeyPressEventArgs^  e)
1225        {
1226            // no permite introducir ningun valor
1227            if (!Char::IsLetterOrDigit(e->KeyChar) || Char::IsLetterOrDigit(e-
    >KeyChar))
1228                e->Handled = true;
1229        }
1230
1231        private: System::Void comboBoxEspecie_SelectedIndexChanged(System::Object^
    sender, System::EventArgs^  e)
1232        {
1233            // si se quiere introducir una especie que no aparece en la lista
1234            if (comboBoxEspecie->SelectedIndex == 2)
1235            {
1236                textBoxEspecie->Enabled = true;
1237                textBoxEspecie->Text = "Especie";
1238                condEspecie = false;
```

```
1239            }
1240            else   // si se escoge una de las especies de la lista
1241            {
1242               textBoxEspecie->Enabled = false;
1243               textBoxEspecie->Text = comboBoxEspecie->Text;
1244            }
1245         }
1246
1247         private: System::Void textBoxEspecie_MouseDown(System::Object^  sender,
     System::Windows::Forms::MouseEventArgs^  e)
1248         {
1249            textBoxEspecie->Text = "";
1250         }
1251
1252         private: System::Void textBoxEspecie_TextChanged(System::Object^  sender,
     System::EventArgs^  e)
1253         {
1254            condEspecie = true;
1255            if (condNombreExp && (condMacho || condHembra) && (condLarva ||
     condAdulto) && (condSagital || condTransversal) && (condIzquierdo || condDerecho)
     && condEspecie && (condLecCB || condLecBC) && condCodigo && condCampana &&
     condBarco && condNumLance && condProfundidad && condFechaLance && (condPesoH ||
     condPesoS) && condPeso && condLongitud && condFechaLec && condNumLec &&
     condNombreInv)
1256               btnOK_InfGral->DialogResult::set(DlgRes::OK);
1257            btnLimpiar_InfGral->Enabled = true;
1258            btnOK_InfGral->Enabled = true;
1259         }
1260
1261         private: System::Void rdBtnLecCB_CheckedChanged(System::Object^  sender,
     System::EventArgs^  e)
1262         {
1263            condLecCB = true;
1264            condLecBC = false;
1265            if (condNombreExp && (condMacho || condHembra) && (condLarva ||
     condAdulto) && (condSagital || condTransversal) && (condIzquierdo || condDerecho)
     && condEspecie && (condLecCB || condLecBC) && condCodigo && condCampana &&
     condBarco && condNumLance && condProfundidad && condFechaLance && (condPesoH ||
     condPesoS) && condPeso && condLongitud && condFechaLec && condNumLec &&
     condNombreInv)
1266               btnOK_InfGral->DialogResult::set(DlgRes::OK);
1267            btnLimpiar_InfGral->Enabled = true;
1268            btnOK_InfGral->Enabled = true;
1269         }
1270
1271         private: System::Void rdBtnLecBC_CheckedChanged(System::Object^  sender,
     System::EventArgs^  e)
1272         {
1273            condLecBC = true;
1274            condLecCB = false;
1275            if (condNombreExp && (condMacho || condHembra) && (condLarva ||
     condAdulto) && (condSagital || condTransversal) && (condIzquierdo || condDerecho)
     && condEspecie && (condLecCB || condLecBC) && condCodigo && condCampana &&
     condBarco && condNumLance && condProfundidad && condFechaLance && (condPesoH ||
     condPesoS) && condPeso && condLongitud && condFechaLec && condNumLec &&
     condNombreInv)
1276               btnOK_InfGral->DialogResult::set(DlgRes::OK);
1277            btnLimpiar_InfGral->Enabled = true;
1278            btnOK_InfGral->Enabled = true;
1279         }
1280
1281         private: System::Void textBoxCodigo_TextChanged(System::Object^  sender,
     System::EventArgs^  e)
1282         {
1283            condCodigo = true;
1284            if (condNombreExp && (condMacho || condHembra) && (condLarva ||
     condAdulto) && (condSagital || condTransversal) && (condIzquierdo || condDerecho)
     && condEspecie && (condLecCB || condLecBC) && condCodigo && condCampana &&
     condBarco && condNumLance && condProfundidad && condFechaLance && (condPesoH ||
     condPesoS) && condPeso && condLongitud && condFechaLec && condNumLec &&
     condNombreInv)
1285               btnOK_InfGral->DialogResult::set(DlgRes::OK);
1286            btnLimpiar_InfGral->Enabled = true;
1287            btnOK_InfGral->Enabled = true;
1288         }
1289
1290         private: System::Void textBoxCampana_TextChanged(System::Object^  sender,
```

```
System::EventArgs^  e)
1291        {
1292           condCampana = true;
1293           if (condNombreExp && (condMacho || condHembra) && (condLarva ||
condAdulto) && (condSagital || condTransversal) && (condIzquierdo || condDerecho)
&& condEspecie && (condLecCB || condLecBC) && condCodigo && condCampana &&
condBarco && condNumLance && condProfundidad && condFechaLance && (condPesoH ||
condPesoS) && condPeso && condLongitud && condFechaLec && condNumLec &&
condNombreInv)
1294              btnOK_InfGral->DialogResult::set(DlgRes::OK);
1295           btnLimpiar_InfGral->Enabled = true;
1296           btnOK_InfGral->Enabled = true;
1297        }
1298
1299        private: System::Void textBoxBarco_TextChanged(System::Object^  sender,
System::EventArgs^  e)
1300        {
1301           condBarco = true;
1302           if (condNombreExp && (condMacho || condHembra) && (condLarva ||
condAdulto) && (condSagital || condTransversal) && (condIzquierdo || condDerecho)
&& condEspecie && (condLecCB || condLecBC) && condCodigo && condCampana &&
condBarco && condNumLance && condProfundidad && condFechaLance && (condPesoH ||
condPesoS) && condPeso && condLongitud && condFechaLec && condNumLec &&
condNombreInv)
1303              btnOK_InfGral->DialogResult::set(DlgRes::OK);
1304           btnLimpiar_InfGral->Enabled = true;
1305           btnOK_InfGral->Enabled = true;
1306        }
1307
1308        private: System::Void textBoxNumLance_KeyPress(System::Object^  sender,
System::Windows::Forms::KeyPressEventArgs^  e)
1309        {
1310           // solo permite numeros y la tecla de retroceso
1311           if (!Char::IsDigit(e->KeyChar) && e->KeyChar != 0x08)
1312              e->Handled = true;
1313        }
1314
1315        private: System::Void textBoxNumLance_TextChanged(System::Object^  sender,
System::EventArgs^  e)
1316        {
1317           condNumLance = true;
1318           if (condNombreExp && (condMacho || condHembra) && (condLarva ||
condAdulto) && (condSagital || condTransversal) && (condIzquierdo || condDerecho)
&& condEspecie && (condLecCB || condLecBC) && condCodigo && condCampana &&
condBarco && condNumLance && condProfundidad && condFechaLance && (condPesoH ||
condPesoS) && condPeso && condLongitud && condFechaLec && condNumLec &&
condNombreInv)
1319              btnOK_InfGral->DialogResult::set(DlgRes::OK);
1320           btnLimpiar_InfGral->Enabled = true;
1321           btnOK_InfGral->Enabled = true;
1322        }
1323
1324        private: System::Void textBoxProfundidad_KeyPress(System::Object^  sender,
System::Windows::Forms::KeyPressEventArgs^  e)
1325        {
1326           // solo permite una coma decimal
1327           if (e->KeyChar == ',')
1328           {
1329              if (textBoxProfundidad->Text->Contains(",") && !textBoxProfundidad-
>SelectedText->Contains(","))
1330                 e->Handled = true;
1331           }
1332           // acepta solo numeros, la coma decimal y la tecla de retroceso
1333           else if (!Char::IsDigit(e->KeyChar) && e->KeyChar != 0x08)
1334              e->Handled = true;
1335        }
1336
1337        private: System::Void textBoxProfundidad_TextChanged(System::Object^
sender, System::EventArgs^  e)
1338        {
1339           condProfundidad = true;
1340           if (condNombreExp && (condMacho || condHembra) && (condLarva ||
condAdulto) && (condSagital || condTransversal) && (condIzquierdo || condDerecho)
&& condEspecie && (condLecCB || condLecBC) && condCodigo && condCampana &&
condBarco && condNumLance && condProfundidad && condFechaLance && (condPesoH ||
condPesoS) && condPeso && condLongitud && condFechaLec && condNumLec &&
condNombreInv)
```

```
1341            btnOK_InfGral->DialogResult::set(DlgRes::OK);
1342         btnLimpiar_InfGral->Enabled = true;
1343         btnOK_InfGral->Enabled = true;
1344      }
1345
1346      private: System::Void dateTimePicker1_KeyPress(System::Object^  sender,
     System::Windows::Forms::KeyPressEventArgs^  e)
1347      {
1348         // no permite introducir ningun valor
1349         if (!Char::IsLetterOrDigit(e->KeyChar) || Char::IsLetterOrDigit(e-
     >KeyChar))
1350            e->Handled = true;
1351      }
1352
1353      private: System::Void dateTimePicker1_ValueChanged(System::Object^  sender,
     System::EventArgs^  e)
1354      {
1355         condFechaLance = true;
1356         if (condNombreExp && (condMacho || condHembra) && (condLarva ||
     condAdulto) && (condSagital || condTransversal) && (condIzquierdo || condDerecho)
     && condEspecie && (condLecCB || condLecBC) && condCodigo && condCampana &&
     condBarco && condNumLance && condProfundidad && condFechaLance && (condPesoH ||
     condPesoS) && condPeso && condLongitud && condFechaLec && condNumLec &&
     condNombreInv)
1357            btnOK_InfGral->DialogResult::set(DlgRes::OK);
1358         btnLimpiar_InfGral->Enabled = true;
1359         btnOK_InfGral->Enabled = true;
1360      }
1361
1362      private: System::Void rdBtnPesoH_CheckedChanged(System::Object^  sender,
     System::EventArgs^  e)
1363      {
1364         condPesoH = true;
1365         condPesoS = false;
1366         if (condNombreExp && (condMacho || condHembra) && (condLarva ||
     condAdulto) && (condSagital || condTransversal) && (condIzquierdo || condDerecho)
     && condEspecie && (condLecCB || condLecBC) && condCodigo && condCampana &&
     condBarco && condNumLance && condProfundidad && condFechaLance && (condPesoH ||
     condPesoS) && condPeso && condLongitud && condFechaLec && condNumLec &&
     condNombreInv)
1367            btnOK_InfGral->DialogResult::set(DlgRes::OK);
1368         btnLimpiar_InfGral->Enabled = true;
1369         btnOK_InfGral->Enabled = true;
1370      }
1371
1372      private: System::Void rdBtnPesoS_CheckedChanged(System::Object^  sender,
     System::EventArgs^  e)
1373      {
1374         condPesoS = true;
1375         condPesoH = false;
1376         if (condNombreExp && (condMacho || condHembra) && (condLarva ||
     condAdulto) && (condSagital || condTransversal) && (condIzquierdo || condDerecho)
     && condEspecie && (condLecCB || condLecBC) && condCodigo && condCampana &&
     condBarco && condNumLance && condProfundidad && condFechaLance && (condPesoH ||
     condPesoS) && condPeso && condLongitud && condFechaLec && condNumLec &&
     condNombreInv)
1377            btnOK_InfGral->DialogResult::set(DlgRes::OK);
1378         btnLimpiar_InfGral->Enabled = true;
1379         btnOK_InfGral->Enabled = true;
1380      }
1381
1382      private: System::Void textBoxPeso_KeyPress(System::Object^  sender,
     System::Windows::Forms::KeyPressEventArgs^  e)
1383      {
1384         // solo permite una coma decimal
1385         if (e->KeyChar == ',')
1386         {
1387            if (textBoxPeso->Text->Contains(",") && !textBoxPeso->SelectedText-
     >Contains(","))
1388               e->Handled = true;
1389         }
1390         // acepta solo numeros, la coma decimal y la tecla de retroceso
1391         else if (!Char::IsDigit(e->KeyChar) && e->KeyChar != 0x08)
1392            e->Handled = true;
1393      }
1394
1395      private: System::Void textBoxPeso_TextChanged(System::Object^  sender,
```

```
System::EventArgs^  e)
1396        {
1397           condPeso = true;
1398           if (condNombreExp && (condMacho || condHembra) && (condLarva ||
condAdulto) && (condSagital || condTransversal) && (condIzquierdo || condDerecho)
&& condEspecie && (condLecCB || condLecBC) && condCodigo && condCampana &&
condBarco && condNumLance && condProfundidad && condFechaLance && (condPesoH ||
condPesoS) && condPeso && condLongitud && condFechaLec && condNumLec &&
condNombreInv)
1399              btnOK_InfGral->DialogResult::set(DlgRes::OK);
1400           btnLimpiar_InfGral->Enabled = true;
1401           btnOK_InfGral->Enabled = true;
1402        }
1403
1404        private: System::Void textBoxLongitud_KeyPress(System::Object^  sender,
System::Windows::Forms::KeyPressEventArgs^  e)
1405        {
1406           // solo permite una coma decimal
1407           if (e->KeyChar == ',')
1408           {
1409              if (textBoxLongitud->Text->Contains(",") && !textBoxLongitud-
>SelectedText->Contains(","))
1410                 e->Handled = true;
1411           }
1412           // acepta solo numeros, la coma decimal y la tecla de retroceso
1413           else if (!Char::IsDigit(e->KeyChar) && e->KeyChar != 0x08)
1414              e->Handled = true;
1415        }
1416
1417        private: System::Void textBoxLongitud_TextChanged(System::Object^  sender,
System::EventArgs^  e)
1418        {
1419           condLongitud = true;
1420           if (condNombreExp && (condMacho || condHembra) && (condLarva ||
condAdulto) && (condSagital || condTransversal) && (condIzquierdo || condDerecho)
&& condEspecie && (condLecCB || condLecBC) && condCodigo && condCampana &&
condBarco && condNumLance && condProfundidad && condFechaLance && (condPesoH ||
condPesoS) && condPeso && condLongitud && condFechaLec && condNumLec &&
condNombreInv)
1421              btnOK_InfGral->DialogResult::set(DlgRes::OK);
1422           btnLimpiar_InfGral->Enabled = true;
1423           btnOK_InfGral->Enabled = true;
1424        }
1425
1426        private: System::Void dateTimePicker2_KeyPress(System::Object^  sender,
System::Windows::Forms::KeyPressEventArgs^  e)
1427        {
1428           // no permite introducir ningun valor
1429           if (!Char::IsLetterOrDigit(e->KeyChar) || Char::IsLetterOrDigit(e-
>KeyChar))
1430              e->Handled = true;
1431        }
1432
1433        private: System::Void dateTimePicker2_ValueChanged(System::Object^  sender,
System::EventArgs^  e)
1434        {
1435           condFechaLec = true;
1436           if (condNombreExp && (condMacho || condHembra) && (condLarva ||
condAdulto) && (condSagital || condTransversal) && (condIzquierdo || condDerecho)
&& condEspecie && (condLecCB || condLecBC) && condCodigo && condCampana &&
condBarco && condNumLance && condProfundidad && condFechaLance && (condPesoH ||
condPesoS) && condPeso && condLongitud && condFechaLec && condNumLec &&
condNombreInv)
1437              btnOK_InfGral->DialogResult::set(DlgRes::OK);
1438           btnLimpiar_InfGral->Enabled = true;
1439           btnOK_InfGral->Enabled = true;
1440        }
1441
1442        private: System::Void textBoxNumLec_KeyPress(System::Object^  sender,
System::Windows::Forms::KeyPressEventArgs^  e)
1443        {
1444           // solo permite numeros y la tecla de retroceso
1445           if (!Char::IsDigit(e->KeyChar) && e->KeyChar != 0x08)
1446              e->Handled = true;
1447        }
1448
1449        private: System::Void textBoxNumLec_TextChanged(System::Object^  sender,
```

```
      System::EventArgs^  e)
1450        {
1451           condNumLec = true;
1452           if (condNombreExp && (condMacho || condHembra) && (condLarva ||
      condAdulto) && (condSagital || condTransversal) && (condIzquierdo || condDerecho)
      && condEspecie && (condLecCB || condLecBC) && condCodigo && condCampana &&
      condBarco && condNumLance && condProfundidad && condFechaLance && (condPesoH ||
      condPesoS) && condPeso && condLongitud && condFechaLec && condNumLec &&
      condNombreInv)
1453              btnOK_InfGral->DialogResult::set(DlgRes::OK);
1454           btnLimpiar_InfGral->Enabled = true;
1455           btnOK_InfGral->Enabled = true;
1456        }
1457
1458        private: System::Void textBoxNombreInv_TextChanged(System::Object^  sender,
      System::EventArgs^  e)
1459        {
1460           condNombreInv = true;
1461           if (condNombreExp && (condMacho || condHembra) && (condLarva ||
      condAdulto) && (condSagital || condTransversal) && (condIzquierdo || condDerecho)
      && condEspecie && (condLecCB || condLecBC) && condCodigo && condCampana &&
      condBarco && condNumLance && condProfundidad && condFechaLance && (condPesoH ||
      condPesoS) && condPeso && condLongitud && condFechaLec && condNumLec &&
      condNombreInv)
1462              btnOK_InfGral->DialogResult::set(DlgRes::OK);
1463           btnLimpiar_InfGral->Enabled = true;
1464           btnOK_InfGral->Enabled = true;
1465        }
1466
1467        private: System::Void textBoxObservaciones_TextChanged(System::Object^
      sender, System::EventArgs^  e)
1468        {
1469           condObservaciones = true;
1470           // no hace comprobaciones porque es opcional
1471           btnLimpiar_InfGral->Enabled = true;
1472           btnOK_InfGral->Enabled = true;
1473        }
1474
1475        // comprobacion final
1476        private: System::Void btnOK_InfGral_Click(System::Object^  sender,
      System::EventArgs^  e)
1477        {
1478           // cadena donde registramos los campos que falten por introducir
1479           String^ errores = "";
1480
1481           if (!condNombreExp)
1482              errores += "Nombre del Experimento\n";
1483           if (!condMacho && !condHembra)
1484              errores += "Género\n";
1485           if (!condLarva && !condAdulto)
1486              errores += "Estado\n";
1487           if (!condSagital && !condTransversal)
1488              errores += "Corte\n";
1489           if (!condIzquierdo && !condDerecho)
1490              errores += "Otolito\n";
1491           if (!condEspecie)
1492              errores += "Especie\n";
1493           if (!condLecCB && !condLecBC)
1494              errores += "Lectura\n";
1495           if (!condCodigo)
1496              errores += "Código\n";
1497           if (!condCampana)
1498              errores += "Campaña\n";
1499           if (!condBarco)
1500              errores += "Barco\n";
1501           if (!condNumLance)
1502              errores += "N° de Lance\n";
1503           if (!condProfundidad)
1504              errores += "Profundidad\n";
1505           if (!condFechaLance)
1506              errores += "Fecha del Lance\n";
1507           if (!condPesoH && !condPesoS)
1508              errores += "Tipo de Peso\n";
1509           if (!condPeso)
1510              errores += "Peso\n";
1511           if (!condLongitud)
1512              errores += "Longitud Total\n";
```

```
1513            if (!condFechaLec)
1514               errores += "Fecha de la Lectura\n";
1515            if (!condNumLec)
1516               errores += "Número de Lecturas\n";
1517            if (!condNombreInv)
1518               errores += "Nombre del Investigador\n";
1519            if (!condObservaciones)
1520               errores += "(opcional: Observaciones)";
1521
1522            // si exinten campos sin introducir
1523            if ((errores != "") && (errores != "(opcional: Observaciones)"))
1524               MessageBox::Show(errores, "Faltan datos por introducir",
    MessageBoxButtons::OK, MessageBoxIcon::Warning);
1525            else   // si todo es correcto
1526            {
1527               MessageBox::Show("Enviando campos a la base de datos", "Información",
    MessageBoxButtons::OK, MessageBoxIcon::Information);
1528               // para acceder despues sin fallos
1529               btnOK_InfGral->Enabled = false;
1530            }
1531            // para volver a empezar
1532            btnOK_InfGral->DialogResult = DlgRes::None;
1533         }
1534
1535         private: System::Void btnLimpiar_InfGral_Click(System::Object^  sender,
    System::EventArgs^  e)
1536         {
1537            result = MessageBox::Show("¿Seguro que desea limpiar\nlos campos del
    formulario?", "Limpiar Base de Datos", MessageBoxButtons::OKCancel,
    MessageBoxIcon::Warning);
1538            if (result == DlgRes::OK)
1539            {
1540               // se vacian los campos
1541               textBoxNombreExp->Text = "";
1542               rdBtnGenM->Checked = false;
1543               rdBtnGenH->Checked = false;
1544               rdBtnEstL->Checked = false;
1545               rdBtnEstA->Checked = false;
1546               rdBtnCorS->Checked = false;
1547               rdBtnCorT->Checked = false;
1548               rdBtnOtoI->Checked = false;
1549               rdBtnOtoD->Checked = false;
1550               comboBoxEspecie->Text = "Especie";
1551               textBoxEspecie->Text = "";
1552               textBoxEspecie->Enabled = false;
1553               rdBtnLecCB->Checked = false;
1554               rdBtnLecBC->Checked = false;
1555               textBoxCodigo->Text = "";
1556               textBoxCampana->Text = "";
1557               textBoxBarco->Text = "";
1558               textBoxNumLance->Text = "";
1559               textBoxProfundidad->Text = "";
1560               dateTimePicker1->Value::set(System::DateTime::Today);
1561               rdBtnPesoH->Checked = false;
1562               rdBtnPesoS->Checked = false;
1563               textBoxPeso->Text = "";
1564               textBoxLongitud->Text = "";
1565               dateTimePicker2->Value::set(System::DateTime::Today);
1566               textBoxNumLec->Text = "";
1567               textBoxNombreInv->Text = "";
1568               textBoxObservaciones->Text = "";
1569
1570               // se resetean las condiciones de comprobacion de errores
1571               condNombreExp = false;
1572               condMacho = false;
1573               condHembra = false;
1574               condLarva = false;
1575               condAdulto = false;
1576               condSagital = false;
1577               condTransversal = false;
1578               condIzquierdo = false;
1579               condDerecho = false;
1580               condEspecie = false;
1581               condLecCB = false;
1582               condLecBC = false;
1583               condCodigo = false;
1584               condCampana = false;
```

```
1585                condBarco = false;
1586                condNumLance = false;
1587                condProfundidad = false;
1588                condFechaLance = false;
1589                condPesoH = false;
1590                condPesoS = false;
1591                condPeso = false;
1592                condLongitud = false;
1593                condFechaLec = false;
1594                condNumLec = false;
1595                condNombreInv = false;
1596                condObservaciones = false;
1597
1598                // se bloquea el boton hasta que vuelva a haber algo que limpiar
1599                btnLimpiar_InfGral->Enabled = false;
1600                // para volver a empezar
1601                btnOK_InfGral->DialogResult = DlgRes::None;
1602                btnOK_InfGral->Enabled = true;
1603
1604                //btnLimpiar_InfGral->DialogResult = DlgRes::Yes;
1605                result = MessageBox::Show("¿Desea salir del diálogo\nInformación
      General?", "Salir", MessageBoxButtons::OKCancel, MessageBoxIcon::Warning);
1606                if (result == DlgRes::OK)
1607                {
1608                   // para limpiar campos de la base de datos
1609                   Dlg_InfGral::DialogResult = DlgRes::Yes;
1610                   // se cierra el formulario
1611                   Dlg_InfGral::Close();
1612                }
1613             }
1614          }
1615 };
1616 }
```

En el constructor del formulario, líneas desde 29 hasta 54, se inicializan todas las variables que se han declarado desde la línea 902 hasta la 927. Son variables booleanas, que se usarán como flags para comprobar si un campo del formulario ha sido introducido de forma correcta o no, para permitir después que se guarden o no en la base de datos.

Después nos encontramos con unas estructuras de programación que no hemos visto antes en nuestro código, son las propiedades para el envío de datos. Al crearse una propiedad pública, la variable que definamos con ella, será visible por otros formularios, en concreto con el Formulario Principal para que luego los envíe a Excel. Así es como funciona este tipo de comunicación. Primero comprobamos que el valor introducido es coherente, y luego, cuando cerramos el formulario porque ya se han rellenado todos los campos, todas las propiedades abren sus puertas para que sean accesibles las variables y puedan almacenarse después. En este formulario, todas las variables de las propiedades públicas son del tipo `String^`, aunque no hay problema en enviar otros tipos, como se verá en el formulario para Calibrar el Microscopio. Pero realmente es más directo así, ya que es muy sencillo enviar cadenas de caracteres a la base de datos.

La singularidad de este formulario radica de la casuística existente a la hora de enviar los datos y de protegerlos frente a errores, porque no es lo mismo evitar que se pueda escribir sobre una lista desplegable, que permitir introducir sólo dígitos en un cuadro de texto que espera un valor numérico, o comprobar si un botón de opción ha sido elegido o no, etc. Por lo tanto, cada variable no sólo necesitará de su propiedad pública

correspondiente para ser enviada, sino que además dispondrá de uno o más manejadores que se encarguen de las pruebas contra fallos.

Para explicar de manera sistemática todos los casos que se abordan en este archivo, vamos a enumerar todos los elementos que se nos pueden presentar:

- Cuadro de texto que espera datos alfanumérico (cualquier letra o dígito, incluyendo símbolos; tipo `String^`).
  - Como `textBoxNombreExp`, que en la línea 941 define su propiedad pública `NombreExp`.
  - En la línea 1132 tiene su manejador del evento de cambio de texto, que se activa con cualquier valor tecleado.
- Cuadro de texto que espera sólo datos numéricos enteros (tipo `int`).
  - Como `textBoxProfundidad`, que en la línea 1054 define su propiedad pública `Profundidad`.
  - En las líneas 1442 y 1449 tiene respectivamente manejadores para evitar introducir nada que no sean números (comprueba cada tecla que se presiona dentro del cuadro, y sólo permite mostrar números o la tecla de retroceso para borrar), y para activarlo cuando se introduzca cualquier dígito (evento de cambio de texto como vimos antes).
- Cuadro de texto que espera sólo datos numéricos decimales (tipo `double`).
  - Como `textBoxNumLec`, que en la línea 1107 define su propiedad pública `NumLec`.
  - En las líneas 1324 y 1337 tiene respectivamente manejadores para evitar introducir nada que no sean números (comprueba cada tecla que se presiona dentro del cuadro, y sólo permite mostrar números, o la coma decimal ",", o la tecla de retroceso para borrar), y para activarlo cuando se introduzca cualquier dígito (evento de cambio de texto como vimos antes).
- Botón de opción que espera ser elegido.
  - Como `rdBtnGenM`, que en la línea 949 define su propiedad pública `Genero`. Pero este elemento se comporta de manera distinta a los anteriores, ya que como de un grupo de botones de opción, sólo uno de ellos saldrá elegido, con una propiedad es suficiente para todos ellos, y sólo hay que comprobar cuál es el que se enviará.
  - Y a la hora de crear el manejador, los botones de opción son más sencillos, gracias al evento de cambio de elección, como observamos en la línea 1144.
- Lista desplegable que espera ser abierta para que se elija una de sus opciones.
  - Como `comboBoxEspecie`, que se apoya en `textBoxEspecie` para enviar su variable `Especie` de propiedad pública en la línea 1001.

    - En las líneas 1224 y 1231 tiene respectivamente manejadores para evitar que se pueda teclear dentro de la lista (evento activado al presionar cualquier tecla), y para cambiar el estado de `textBoxEspecie` dependiendo de la opción que se escoja (evento de cambio de opción elegida).
- Cuadro de fecha
    - Como `dateTimePicker1`, que en la línea 1062 define su propiedad pública `FechaLance`.
    - En las líneas 1346 y 1353 tiene manejadores que realizan la misma función que los de `comboBoxEspecie`, no dejar escribir nada dentro del cuadro (evento activado al presionar cualquier tecla), y activarse cuando se cambie su estado (evento de cambio de valor).

Y acerca de los botones, cada vez que se rellena correctamente un campo del formulario, se valida una condición para que al pulsar Aceptar se envíen los datos, y también se habilita el botón Limpiar para borrarlo todo, permaneciendo inactivo mientras no se escriba nada. Las dos últimas funciones de eventos, en las líneas 1476 para Aceptar y 1535 para Limpiar, son para que al hacer clic sobre estos dos botones todo reaccione como debe. Si hemos rellenado correctamente todos los campos y pulsamos Aceptar, el valor devuelto por Información General al Formulario Principal es el indicado para guardar los datos en la base de datos, mientras que si pulsamos Limpiar, otro valor será devuelto al Formulario Principal, y éste borrará lo que se hubiera almacenado en el documento de Excel.

Para que esto funcione así, es necesario hacer un paso intermedio, porque por defecto, el botón Aceptar devuelve un valor concreto al ser pulsado, pero nosotros hemos querido hacer una comprobación en medio. Por ejemplo, si el botón Aceptar está configurado como el "Botón Aceptar" del formulario de Información General, aunque tuviéramos campos sin rellenar, al pulsar Aceptar se cerraría el formulario, dejando incompleto el proceso de captación de datos. Por ello hay que anular el valor que devuelve este botón, y activarlo sólo cuando realmente sea preciso, que es cuando el formulario ha sido completado. Para concretar ideas, tenemos que cambiar la propiedad "DialogResult" de "OK" a "None", para que no haga nada al ser pulsado, o al menos que no haga nada por defecto, y ya seremos nosotros los que digamos lo que tiene que hacer en cada caso. Para diferenciar el resultado devuelto por el botón Limpiar, se eligió que devolviera "Yes". Entonces si el Formulario Principal obtiene como valor devuelto de Información General el comando "OK", está listo para que al pulsar el botón Guardar, todo vaya a Excel. Algo semejante pasa con el valor devuelto "Yes", con el que se borran los campos de la base de datos.