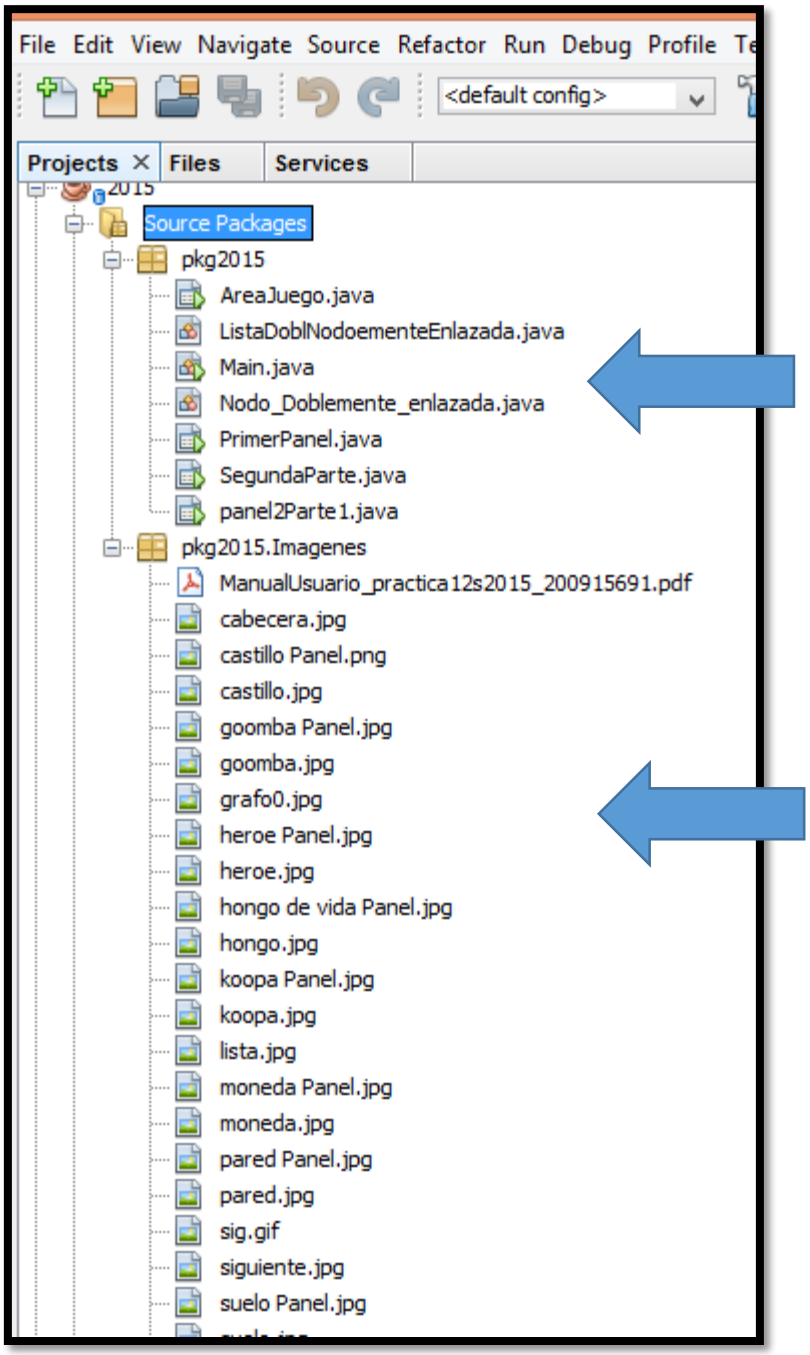
MANUAL TECNICO

En este manual técnico podrá encontrar el modo de uso de Mario Maker Guatemalteco, y como poder actuar ante inconvenientes o para cuando se desee agregar alguna funcionalidad a la aplicación.

Antes debes saber la distribución del proyecto, para poder actuar en base a estandarización.



Sección donde podremos agregar las clases que necesitemos y donde se encuentran las clases actuales con la lógica de la aplicación.

Podremos agregar imágenes complementos de cosas que nos puedan servir, en el desarrollo de la aplicación.

A continuación daremos a conocer las funcionas más importantes de la aplicación.

CLASE NODO

```
public class ListaDoblNodoementeEnlazada {
    String Nombre;
    Integer Id;

    ListaDoblNodoementeEnlazada ant,sig;

   private ListaDoblNodoementeEnlazada raiz;

    public ListaDoblNodoementeEnlazada () {
        raiz=null;
    }
```

En esta clase definiremos nuestra clase que nos servirá para manejar, la lógica e información de nuestra información.

Método Borrar:

```
   public void borrar (int pos)
   {

       if (pos <= cantidad ())     {
           if (pos == 1) {
               raiz = raiz.sig;
               if (raiz!=null)
                   raiz.ant=null;
           } else {
               ListaDoblNodoementeEnlazada reco;
               reco = raiz;
               for (int f = 1 ; f <= pos - 2 ; f++)
                   reco = reco.sig;
               ListaDoblNodoementeEnlazada prox = reco.sig;
               prox=prox.sig;
               reco.sig = prox;
               if (prox!=null)
                   prox.ant=reco;
           }
       }
   }
```

Este método nos permitirá poder borrar elementos de nuestra lista y de nuestra colección doblemente enlazada

CLASE INSERTAR

```
public void insertar (Integer pos, Integer x, String Nombre)
{
    if (pos <= cantidad () + 1)    {
        ListaDoblNodoementeEnlazada nuevo = new ListaDoblNodoementeEnlazada ();
        nuevo.Id = x;
        nuevo.Nombre = Nombre;
        if (pos == 1){
            nuevo.sig = raiz;
            if (raiz!=null)
                raiz.ant=nuevo;
            raiz = nuevo;
        } else
            if (pos == cantidad () + 1)    {
                ListaDoblNodoementeEnlazada reco = raiz;
                while (reco.sig != null) {
                    reco = reco.sig;
                }
                reco.sig = nuevo;
                nuevo.ant=reco;
                nuevo.sig = null;

            } else {
                ListaDoblNodoementeEnlazada reco = raiz;
                for (int f = 1 ; f <= pos - 2 ; f++)
                    reco = reco.sig;
                ListaDoblNodoementeEnlazada siguiente = reco.sig;
                reco.sig = nuevo;
                nuevo.ant=reco;
                nuevo.sig = siguiente;
                siguiente.ant=nuevo;
            }

    }
}
```

Este método nos permitirá poder ir agregando elementos de tipo Nodo a nuestra colección.

Creación de Variables de conteo.

```
public class PrimerPanel extends javax.swing.JFrame {
 public static ListaDoblNodoementeEnlazada  Nodo = new ListaDoblNodoementeEnlazada();
  public static Integer Contador = 1;
 public static Integer banderaGeneral=0;
 public static Integer contadorHeore=0;
 public static Integer contadorCastillo=0;
 public static Integer contadorMuro=0;
 public static Integer contadorSuelo=0;
 public static Integer contadorGoomba=0;
 public static  Integer contadorHongoVida=0;
 public static  Integer contadorMoneda=0;
 public static Integer contadorCKoopa=0;
  public static String Nodo1 = "" ;
```

Cada una de estas variables nos servirá de contadores para el flujo dentro del proyecto. Cada nombre representa lo que esta variable identificara.

METODO PARA TRASLADO ENTRE PANTALLAS

```
panel2Parte1 dg = null;
                try {

                        dg = new panel2Parte1();
                        } catch (Exception ex) {
                        Logger.getLogger(PrimerPanel.class.getName()
                        ).log(Level.SEVERE, null, ex);
                        }
                        dg.setLocationRelativeTo(null);
                        dg.setVisible(true);
                        this.setVisible(false);
```

Este método nos permite trasladarnos entre las pantallas, en el cual dg significa la pantalla de donde estamos y en él .getLogger indicamos hacia la pantalla que queremos movernos.

Método para pasar valores entre clase.

```
   public Integer getContadorMondeda(){
return this.contadorMoneda;
}
```

Este método deber ser aplicable con cada uno de los valores que deseamos pasar entre las clases de java.

Método para crear Grafos:

```
public void setearImagenPanel(){
String total=this.generarTxtGrafo();
this.EliminarArchivo("C:\\Users\\J.Y.N.CH\\Desktop\\grafo1.txt");
this.txt(total, "C:\\Users\\J.Y.N.CH\\Desktop\\grafo1.txt");
this.grafo("C:\\Users\\J.Y.N.CH\\Desktop\\grafo1.txt","C:\\Users\\J.Y.N.CH\\Documents\\NetBeansProjects\\2015\\src\\pkg2015\\Imagenes\\g
this.prueba++;
}
```

Este método resume los métodos necesarios para poder agregar grafos.