



TP N° 4: “WIKI CRAFTEO”

Celeste Cisternas 2A



22/11/2021

UTN

Form Principal: Wiki Crafteo



1- CONSEGUIR BLOQUES: botón que abre un formulario para conseguir bloques de 3 tipos, madera, piedra o diamante.

2- CONSTRUIR HERRAMIENTAS: botón que abre un formulario para construir herramientas, se puede acceder solo si antes consiguió bloques. Además cada vez que construye una herramienta se le descuenta un bloque, el bloque utilizado determina el tipo de material con el que esta hecha la herramienta.

3- ESTADÍSTICAS: botón que abre un formulario en el cual se encuentran los porcentajes y las descargas de los mismos.

4- VER LISTA: botón que abre un formulario con un datagrid para ver la lista de jugadores con sus inventarios. Aca se puede borrar o editar jugadores e incluso vaciar los inventarios.

6- SALIR: botón que cierra el formulario completamente y termina la aplicación.

Antes de poder ingresar a los primeros dos botones, se le solicita al usuario un registro. Deben colocar su nombre de usuario, si no se encontraban registrados se añaden a la lista y si ya se encontraban, se abre el formulario con el inventario que le corresponde al jugador.



PRIMER BOTON: CONSEGUIR BLOQUES



Formulario que permite elegir el material del bloque a solicitar y su cantidad, tiene un label que avisa de la cantidad de espacio de inventario, que va disminuyendo o aumentando dependiendo de los movimientos que haga el usuario. Al guardar, se guardan los datos en el inventario privado del jugador, que tiene un máximo de 20 de capacidad. En caso de llegar al límite del inventario, se lanza un evento de CapacidadMaxima y se le da la opción al usuario de vaciar el inventario en ese mismo momento.


SEGUNDO BOTON: FORMULARIO CONSTRUIR HERRAMIENTAS

Antes de ingresar, el jugador debe tener bloques en su inventario ya que cada herramienta construida depende del bloque.

El formulario construir herramientas te permite elegir entre tres tipos, pico, hacha y espada. Además el jugador debe elegir el bloque que tiene disponible, para eso se le muestra debajo de cada tipo de material, la cantidad de bloques que tiene disponible. Se debe seleccionar una cantidad acorde al espacio del inventario mostrado en el label superior de lo contrario se lanzará una excepción.

Crafting

Espacio inventario: 16

		
<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="4"/>

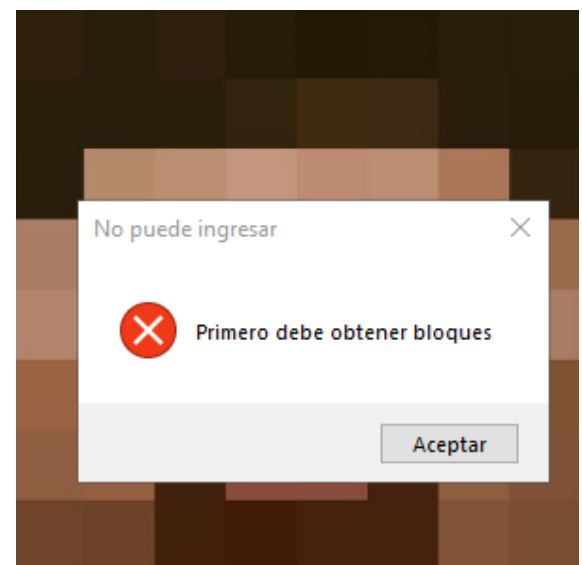
Cantidad de bloques disponibles:

CONSTRUIR

0

SALIR

Seleccionar ▾



TERCER BOTON: FORMULARIO ESTADISTICAS

En este formulario se encuentra la implementacion de archivos txt,xml y json. primero se debe elegir en el combobox que tipo de extension tendra su archivo y luego se selecciona uno de los porcentajes. Se abrira una ventanita de descarga, realizada con un hilo secundario que muestra el porcentaje de descarga del archivo y luego se guardara en una carpeta en el escritorio "Informes WikiCrafteo".



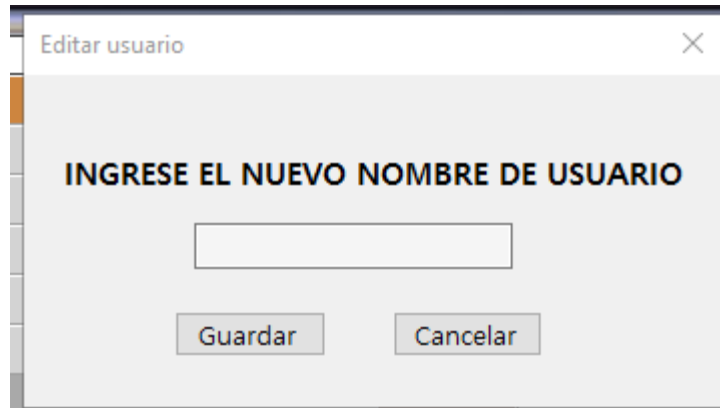
CUARTO BOTON: VER LISTA

Contiene un datagrid en el cual se cargan los jugadores de la base de datos, al seleccionar un jugador de la lista y tocar "ver inventario" sale en un costado la lista de cada jugador. Si se deseara borrar uno, se lo debe seleccionar y tocar "borrar"



lo cual lo elimina de la lista y de la base de datos, sin perder la correlatividad del id cuando se desee agregar uno nuevo, se produce una baja logica.

Si se selecciona un jugador y luego editar, se abre una ventana que solicita que se ingrese un nuevo nombre, esto permite modificar el nombre de usuario del jugador.



APLICACIÓN DE LOS TEMAS VISTOS

EXCEPCIONES:

```
3 referencias
public class CantidadIncorrectaException : Exception
{
    static string msj = "Elija una cantidad correcta";
    2 referencias
    public CantidadIncorrectaException() : base(msj)
    {
    }
}
```

```
        else
        {
            if (cantidad == 0 || cantidad > capacidadInventario)
            {
                throw new CantidadIncorrectaException();
            }
        }
    }
    catch (SinSeleccionException ex)
    {
        MessageBox.Show(ex.Message);
    }
}
catch (Exception ex)
{
    MessageBox.Show(ex.Message);
}
```

PRUEBAS UNITARIAS:

```
[TestClass]
0 referencias
public class UnitTest1
{
    /// <summary>
    /// Prueba de pasarle un nombre de usuario y que busque si se encuentra en la lista de jugadores, de no estar, crea uno nuevo con ese usuario
    /// si el jugador es nuevo, devuelve true para generar el id
    /// </summary>
    [TestMethod]
    0 referencias
    public void GetJugador_CuandoEncuentraElUsuario0no_DeberiaRetornarElJugador()
    {
        //Arrange
        Wiki lista = new Wiki();
        string usuario = "celeste22";
        Jugador player = new Jugador(1, usuario);
        lista.Jugadores.Add(player);
        Jugador jugadorRecibido;
        bool isNew;

        //Act
        jugadorRecibido = lista.GetJugador(usuario, out isNew);

        //Assert
        Assert.AreEqual(usuario, jugadorRecibido.Usuario); //compara el nombre entregado al metodo con el nombre del jugador que devuelve el metodo
        Assert.IsFalse(isNew); //si es falso el jugador es nuevo
    }
}
```

TIPOS GENERICOS:

INTERFACES:

```
1 referencia
public interface IHerramienta
{
    2 referencias
    string Uso(ETipoHerramienta tipo);

    2 referencias
    int DurabilidadPorTipo(ETipoMaterial material);
}
```

ARCHIVOS Y SERIALIZACION:

```
1 referencia
public static bool SerializarXml(string archivo, Wiki wiki)
{
    bool isOk = false;
    try
    {
        using (XmlTextWriter writer = new XmlTextWriter(archivo, Encoding.UTF8))
        {
            XmlSerializer serializer = new XmlSerializer(typeof(Wiki));

            serializer.Serialize(writer, wiki);
            isOk = true;
        }
    }
    catch (Exception)
    {
        throw;
    }
    return isOk;
}
```

SQL:

```
0 referencias
static JugadorAccesoDatos()
{
    connectionString = @"Data Source=DESKTOP-Q90H3PF;Database=WikiCrafteoDB;Trusted_Connection=True;";
    command = new SqlCommand();
    connection = new SqlConnection(connectionString);
    command.Connection = connection;
    command.CommandType = CommandType.Text;
}

3 referencias
public static DataTable LeerConsulta()
{
    DataTable dt = new DataTable();
    try
    {
        if (connection.State != ConnectionState.Open)
        {
            connection.Open();
        }

        SqlCommand cmd = new SqlCommand("consulta_ObtenerTodos", command.Connection);
        cmd.CommandType = CommandType.StoredProcedure;

        SqlDataAdapter sd = new SqlDataAdapter(cmd);
        sd.Fill(dt);

        cmd.ExecuteReader();

        return dt;
    }
    catch (Exception)
    {
        throw;
    }
    finally
    {
        connection.Close();
    }
}
}
```

DELEGADOS Y EXPRESIONES LAMBDA E HILOS:

```
1 referencia
private void btn_Inventario_Click(object sender, EventArgs e)
{
    try
    {
        if (dtgv_VerLista.SelectedRows.Count > 0)
        {
            Jugador player = (Jugador)dtgv_VerLista.CurrentRow.DataBoundItem;
            Task hilo = Task.Run(() => TraerContenido(rtb_Inventario, player));
        }
    }
    catch (Exception)
    {
        throw;
    }
}

3 referencias
public void TraerContenido(RichTextBox textBox, Jugador jugador)
{
    if (InvokeRequired)
    {
        Action<RichTextBox, Jugador> action = TraerContenido;
        object[] parameters = new object[] { textBox, jugador };
        this.Invoke(action, parameters);
    }
    else
    {
        textBox.Text = jugador.Inventario.ToString();
    }
}
}
```


EVENTOS:

```
if (i.Capacidad <= 1)
{
    if (i.EventoCapacidad != null)
    {
        i.EventoCapacidad(i, EventArgs.Empty);
        isOk = false;
    }
}
```

```
1 referencia
private void Inventario_EventoCapacidad(object sender, EventArgs e)
{
    if (MessageBox.Show("Le queda poco espacio ¿Desea vaciar el inventario?", "LIMITE CAPACIDAD", MessageBoxButtons.OKCancel, MessageBoxIcon.Information) == DialogResult.OK)
    {
        ((Inventario)sender).ListaElementos.Clear();
        ((Inventario)sender).Capacidad = 20;
        eve = true;
    }
    else
    {
        MessageBox.Show("Inventario completo", "LIMITE CAPACIDAD", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}
```

METODOS DE EXTENSION: (continua, solo puse un pedazo)

```
3 referencias
public static bool ImprimirPorcentaje(this Wiki lista, int boton, string seleccion)
{
    bool isOk = true;
    double porcentaje = 0;
    string tipo = "";
    string nombre = "";

    switch (boton)
    {
        case 1:
            porcentaje = lista.PorcentajeConInventarioLleno();
            tipo = "LLENO";
            nombre = @"Porcentaje Inventario lleno";
            break;
        case 2:
            porcentaje = lista.PorcentajeDeDiamanteEnInventarios();
            tipo = "DE DIAMANTE";
            nombre = @"Porcentaje Inventario de diamantes";
            break;
        case 3:
            porcentaje = lista.PorcentajeDePiedraEnInventarios();
            tipo = "DE PIEDRA";
            nombre = @"Porcentaje Inventario de piedra";
            break;
        case 4:
            porcentaje = lista.PorcentajeDeMaderaEnInventarios();
            tipo = "DE MADERA";
            nombre = @"Porcentaje Inventario de madera";
            break;
    }
}
```