

TÍTULO

DESARROLLO DE APLICACIONES I

Sesión Nro. 12



Objetivos de la sesión:

Al culminar la presente podrás:

- Implementar el mantenimiento de una tabla desde formularios ASP. Net reutilizando las capas funcionales ADO, BL y el proyecto de entidades de negocio BE.
- Interacción con ModalPopup AJAX para agilizar las ventanas de inserción y actualización de registros.
- Implementar una paginación local para mejor presentación de los registros en la vista de listado.



Temario

1. Adaptación de la plantilla de mantenimiento
2. Codificación de los principales eventos
3. Paginación del control GridView



Tema Nro. 1: Adaptación de la plantilla de mantenimiento



1.1 Adaptar el diseño de la plantilla

- Como parte de la implementación del mantenimiento de la tabla de proveedores (basándonos en lo hecho en la capa de presentación Windows) en la capa de presentación WEB, se deberá adaptar la plantilla proporcionada por su instructor, de tal manera que se tenga el siguiente diseño final



1.2 Diseño final WebMantProveedores: Nuevo Proveedor

MANTENIMIENTO DE PROVEEDORES

Nuevo proveedor

Ingrese razon social:

Codigo	Razon Social	Direccion	Telefono	Departamento	Provincia	Distrito	Estado
V137	ACABADOS IR	CALLE JULIOCESAR 723		AREQUIPA	CAYLLOMA	LLUTA	Activo
V005	Acker	Calle La...	545				
V192	ALBRESA AGUAS SAC						
V025	BACKUS & JOHNSTON UNION DE CERVECERIAS	AV. NIO 3986					
V108	BLAVES INTERNACIONAL SAC	C.LAS B 482 - B					
V094	BUEN AMIGO SAC	CALLE 321					
V153	CASTOR -INTERFOREST SAC	REP. PA					
V117	COMERCIAL YARMAS SAC	AV MAR					
V180	COMP SER SAC						
V119	CONTINENTAL SAC	CALLE SAN PEDRO 671		LIMA	LIMA	SAN LUIS	Activo
V093	CORPORACION JULISSA S.A.	CALLE SAN FERNANDO 781		LIMA	LIMA	SAN BARTOLO	Activo
V154	COSPORACION BEINT SAC			AREQUIPA	CAYLLOMA	LLUTA	Activo
V143	CREACIONES HUANCA	JR. GAMARRA 901		AREQUIPA	AREQUIPA	SAN JUAN DE TABUCANT	Activo

Nuevo Proveedor

Razon social:

Direccion:

Telefono:

RUC:

Departamento:

Provincia:

Distrito:

Representante de Ventas:

Estado: ☒



1.2 Diseño final WebMantProveedores: Actualizar Proveedor

MANTENIMIENTO DE PROVEEDORES

Ingrese razon social:

Codigo	Razon Social	Direccion	Telefono	Departamento	Provincia	Distrito	Estado
V137	ACABADOS IR	CALLE JULIOCESAR 723		AREQUIPA	CAYLLOMA	LLUTA	Activo
V005	Acker	Calle La	545				
V192	ALBRESA AGUAS SAC						
V025	BACKUS & JOHNSTON UNION DE CERVECERIAS	AV. NIO 3986					
V108	BLAVES INTERNACIONAL SAC	C.LAS B 482 - B					
V094	BUEN AMIGO SAC	CALLE I 321					
V153	CASTOR -INTERFOREST SAC	REP. PA					
V117	COMERCIAL YARMAS SAC	AV MAR					
V180	COMPSE SAC						
V119	CONTINENTAL SAC	CALLE SAN PEDRO 673		LIMA	LIMA	CAN LUC	Activo

Actualizar Proveedor

Codigo:

Razon social:

Direccion:

Telefono:

RUC:

Departamento:

Provincia:

Distrito:

Representante de Ventas:

Estado: ☒



Tema Nro. 2: Codificación de los principales eventos



2.1 Evento Load

- En este evento de cargara el grivView con los datos filtrados por razón social

```
public partial class WebMantProveedores : System.Web.UI.Page
{
    // Declaramos las instancias ...
    ProveedorBL objProveedorBL = new ProveedorBL();
    ProveedorBE objProveedorBE = new ProveedorBE();
    UbigeoBL objUbigeoBL = new UbigeoBL();
    DataView dtv;

    protected void Page_Load(object sender, EventArgs e)
    {
        try
        {
            if (Page.IsPostBack == false)
            {
                // Creamos la vista en memoria y cargamos los datos
                dtv = new DataView(objProveedorBL.ListarProveedor());
                Session["Vista"] = dtv;
                CargarDatos("");
            }
        }
        catch (Exception ex)
        {
            lblMensaje.Text = ex.Message;
            mpeMensaje.Show();
        }
    }
}
```

```
private void CargarDatos(String strFiltro)
{
    dtv = (DataView)Session["Vista"];
    dtv.RowFilter = "raz_soc_prv like '%" + strFiltro + "%'";
    grvProveedor.DataSource = dtv;
    grvProveedor.DataBind();

    if (grvProveedor.Rows.Count == 0)
    {
        lblMensaje.Text = "No existen registros con el criterio ingresado.";
        mpeMensaje.Show();
    }
}
```



2.2 El evento clic del botón Nuevo

- Para insertar un nuevo proveedor debemos preparar el modal popup PopMant, limpiando los controles y cargando el ubigeo.

```
protected void btnNuevo_Click(object sender, EventArgs e)
{
    try
    {
        // Limpiamos todos los controles del panel1 asociados al modalpopup
        // de insercion (PopMant)
        lblMensajeIns.Text = "";
        txtRS.Text = "";
        txtDir.Text = "";
        txtRUC.Text = "";
        txtTel.Text = "";
        txtRepVen.Text = "";
        // Activo por defecto al insertar
        chkActivo.Checked = true;
        // Cargamos el ubigeo para insertar el nuevo proveedor
        CargarUbigeo("14", "01", "01"); // Lima, Lima , Lima por defecto
        // Mostramos el PopMant
        txtRS.Focus();
        PopMant.Show();
    }
    catch (Exception ex)
    {
        lblMensaje.Text = ex.Message;
        mpeMensaje.Show();
    }
}
```

```
private void CargarUbigeo(String IdDepa, String IdProv, String IdDist)
{
    // Combos de ubigeo para la insercion del proveedor
    cboDepartamento.DataSource = objUbigeoBL.Ubigeo_Departamentos();
    cboDepartamento.DataValueField = "IdDepa";
    cboDepartamento.DataTextField = "Departamento";
    cboDepartamento.DataBind();
    cboDepartamento.SelectedValue = IdDepa;
    cboProvincia.DataSource = objUbigeoBL.Ubigeo_ProvinciasDepartamento(IdDepa);
    cboProvincia.DataValueField = "IdProv";
    cboProvincia.DataTextField = "Provincia";
    cboProvincia.DataBind();
    cboProvincia.SelectedValue = IdProv;
    cboDistrito.DataSource = objUbigeoBL.Ubigeo_DistritosProvinciaDepartamento
                                                                    (IdDepa, IdProv);
    cboDistrito.DataValueField = "IdDist";
    cboDistrito.DataTextField = "Distrito";
    cboDistrito.DataBind();
    cboDistrito.SelectedValue = IdDist;
}
```

2.3 El evento clic del botón Grabar

- Para grabar el nuevo proveedor debemos configurar la entidad de negocio y llamar al método Insertar

```
protected void btnGrabar_Click(object sender, EventArgs e)
{
    try
    {
        // Puede hacer las validaciones por codigo (o agregar RequiredFieldValidator)
        // Como ejemplo solo validaremos la Razon Social
        if (txtRS.Text == "")
        {
            throw new Exception("La razon social es obligatoria.");
        }

        // Tras las validaciones se configura la instancia de objProveedorBE y se invoca al metodo Insertar
        // del objProveedorBL

        objProveedorBE.Raz_soc_prv = txtRS.Text;
        objProveedorBE.Dir_prv = txtDir.Text;
        objProveedorBE.Ruc_prv = txtRUC.Text;
        objProveedorBE.Tel_prv = txtTel.Text;
        objProveedorBE.Rep_ven = txtRepVen.Text;
        objProveedorBE.Id_Ubigeo = cboDepartamento.SelectedValue +
                                cboProvincia.SelectedValue + cboDistrito.SelectedValue;
        objProveedorBE.Usu_Registro = "jleon";
        objProveedorBE.Est_prv = Convert.ToInt16(chkActivo.Checked);

        // Insertamos...
        if (objProveedorBL.InsertarProveedor(objProveedorBE) == true)
        {
            dtv = new DataView(objProveedorBL.ListarProveedor());
            Session["Vista"] = dtv;
            CargarDatos(txtFiltro.Text);
        }
    }
}
```

2.3 El evento clic del botón Grabar

- Para grabar el nuevo proveedor debemos configurar la entidad de negocio y llamar al método Insertar

```
    else
    {
        throw new Exception("No se inserto el registro. Contacte con IT.");
    }
}
catch (Exception ex)
{
    lblMensajeIns.Text = ex.Message;
    PopMant.Show();// para mantener el popup abierto tras el clic en Grabar
}
}
```

2.4 El evento RowCommand del GridView

- Para actualizar un proveedor debemos indicar cual es, haciendo clic en el el ColumnButton “Editar” del grid y mostrando en el PopMant2 el proveedor seleccionado

```
protected void grvProveedor_RowCommand(object sender, GridViewCommandEventArgs e)
{
    try
    {
        // Se obtiene en que fila del grid se ha seleccionado el boton
        Int16 fila = Convert.ToInt16(e.CommandArgument);
        // Validamos si el boton tiene el nombre Editar
        if (e.CommandName == "Editar")
        {
            // Limpiamos el lblMensajeAct
            lblMensajeAct.Text = "";
            // Obtenemos el codigo del proveedor de la celda 1 de la fila seleccionada
            String strCod = grvProveedor.Rows[fila].Cells[1].Text;
            // Obtenemos la instancia del proveedor a actualizar
            objProveedorBE = objProveedorBL.ConsultarProveedor(strCod);
            // Mostramos el proveedor en los controles del panel del PopMant2
            // (popup para actualizar registro)
            txtCod.Text = objProveedorBE.Cod_prv;
            txtRS2.Text = objProveedorBE.Raz_soc_prv;
            txtDir2.Text = objProveedorBE.Dir_prv;
            txtRUC2.Text = objProveedorBE.Ruc_prv;
            txtTel2.Text = objProveedorBE.Tel_prv;
            txtRepVen2.Text = objProveedorBE.Rep_ven;
            chkActivo2.Checked = Convert.ToBoolean(objProveedorBE.Est_prv);
            String Id_Ubigeo = objProveedorBE.Id_Ubigeo;
            //Manejamos el ubigeo para el panel de PopMant2
            CargarUbigeo2(Id_Ubigeo.Substring(0, 2), Id_Ubigeo.Substring(2, 2),
                Id_Ubigeo.Substring(4, 2));

            //Mostramos el PopMant2
            PopMant2.Show();
        }
    }
}
```

```
catch (Exception ex)
{
    lblMensaje.Text = ex.Message;
    mpeMensaje.Show();
}

private void CargarUbigeo2(String IdDepa, String IdProv, String IdDist)
{
    // Combos de ubigeo para la insercion del proveedor
    cboDepartamento2.DataSource = objUbigeoBL.Ubigeo_Departamentos();
    cboDepartamento2.DataValueField = "IdDepa";
    cboDepartamento2.DataTextField = "Departamento";
    cboDepartamento2.DataBind();
    cboDepartamento2.SelectedValue = IdDepa;
    cboProvincia2.DataSource = objUbigeoBL.Ubigeo_ProvinciasDepartamento(IdDepa);
    cboProvincia2.DataValueField = "IdProv";
    cboProvincia2.DataTextField = "Provincia";
    cboProvincia2.DataBind();
    cboProvincia2.SelectedValue = IdProv;
    cboDistrito2.DataSource = objUbigeoBL.Ubigeo_DistritosProvinciaDepartamento(IdDepa, IdProv);
    cboDistrito2.DataValueField = "IdDist";
    cboDistrito2.DataTextField = "Distrito";
    cboDistrito2.DataBind();
    cboDistrito2.SelectedValue = IdDist;
}
```

2.5 El evento clic del botón btnGrabar2

- Si queremos actualizar los datos del proveedor debemos codificar el botón btnGrabar2 del PopMant

```
protected void btnGrabar2_Click(object sender, EventArgs e)
{
    try
    {
        // Tras las validaciones se configura la instancia de objProveedorBE y se invoca al metodo Actualizar
        // del objProveedorBL
        objProveedorBE.Cod_prv = txtCod.Text;
        objProveedorBE.Raz_soc_prv = txtRS2.Text;
        objProveedorBE.Dir_prv = txtDir2.Text;
        objProveedorBE.Ruc_prv = txtRUC2.Text;
        objProveedorBE.Tel_prv = txtTel2.Text;
        objProveedorBE.Rep_ven = txtRepVen2.Text;
        objProveedorBE.Id_Ubigeo = cboDepartamento2.SelectedValue + cboProvincia2.SelectedValue + cboDistrito2.SelectedValue;
        objProveedorBE.Usu_Ult_Mod = "jleon";
        objProveedorBE.Est_prv = Convert.ToInt16(chkActivo2.Checked);
        // Actualizamos...
        if (objProveedorBL.ActualizarProveedor(objProveedorBE) == true)
        {
            dtv = new DataView(objProveedorBL.ListarProveedor());
            Session["Vista"] = dtv;
            CargarDatos(txtFiltro.Text);
        }
        else
        {
            throw new Exception("No se actualizo el registro. Contacte con IT.");
        }
    }
    catch (Exception ex)
    {
        lblMensajeAct.Text = ex.Message;
        PopMant2.Show();
    }
}
```


2.6 Eventos adicionales

- No olvidarse los eventos `SelectedIndexChanged` de los combos de ubigeo `cboDepartamento`, `cboDepartamento2`, `cboProvincia` y `cboProvincia2` (que deben tener la propiedad `AutopostBack` en `true`).

```
protected void cboDepartamento_SelectedIndexChanged(object sender, EventArgs e)
{
    CargarUbigeo(cboDepartamento.SelectedValue.ToString(), "01", "01");
    PopMant.Show(); //Para que al postear el evento SelectedIndexChanged se mantenga abierto el PopMant
}
```

```
protected void cboProvincia_SelectedIndexChanged(object sender, EventArgs e)
{
    CargarUbigeo(cboDepartamento.SelectedValue.ToString(), cboProvincia.SelectedValue.ToString(), "01");
    PopMant.Show(); //Para que al postear el evento SelectedIndexChanged se mantenga abierto el PopMant
}
```

```
protected void cboDepartamento2_SelectedIndexChanged(object sender, EventArgs e)
{
    CargarUbigeo2(cboDepartamento2.SelectedValue.ToString(), "01", "01");
    PopMant2.Show(); //Para que al postear el evento SelectedIndexChanged se mantenga abierto el PopMant2
}
```

```
protected void cboProvincia2_SelectedIndexChanged(object sender, EventArgs e)
{
    CargarUbigeo2(cboDepartamento2.SelectedValue.ToString(), cboProvincia2 .SelectedValue .ToString (), "01");
    PopMant2.Show(); //Para que al postear el evento SelectedIndexChanged se mantenga abierto el PopMant2
}
```

2.7 Eventos adicionales

- Por ultimo el evento clic del botón btnFiltrar para aplicar el filtro por razón social y el evento RowDataBound del GridView para pintar de rojo los proveedores inactivos

```
protected void btnFiltrar_Click(object sender, ImageClickEventArgs e)
{
    try
    {
        CargarDatos(txtFiltro.Text);
    }
    catch (Exception ex)
    {
        lblMensaje.Text = ex.Message;
        mpeMensaje.Show();
    }
}
```

```
protected void grvProveedor_RowDataBound(object sender, GridViewRowEventArgs e)
{
    try
    {
        // El evento recorre cada fila del grid cada vez que este se enlaza a su origen de datos
        if (e.Row.RowType == DataControlRowType.DataRow)
        {
            // Obtenemos el valor del campo Estado de cada fila (celda 8 )
            String estado = e.Row.Cells[8].Text;
            // Si esta inactivo el color de la fuente (ForeColor) de la fila estara en rojo
            if (estado=="Inactivo")
            {
                e.Row.ForeColor = System.Drawing . Color.Red;
            }
        }
    }
    catch (Exception ex)
    {
        lblMensaje.Text = ex.Message;
        mpeMensaje.Show();
    }
}
```


Tema Nro. 3: La paginación de datos en un GridView



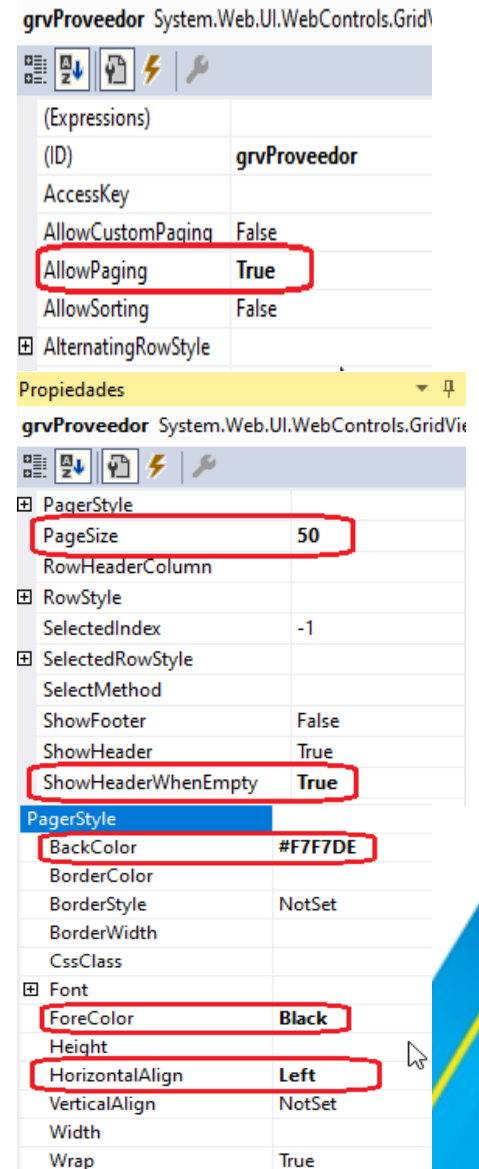
3.1 Paginando en un GridView

- Cuando hay muchos registros que mostrar en un GridView lo mas recomendable es paginarlos, para que se muestren dichos registros en paginas de 20 en 20 (recomendable) o de 50 en 50.
- Hay 2 formas de hacer esto: paginando el GridView o paginando la consulta desde un store procedure.
- En el mantenimiento de Proveedores emplearemos la primera manera, que si bien es cierto no es la mas eficiente, para volúmenes de datos menores a 1000 es aceptable.
- Veremos en un próximo ejemplo la paginación desde Store Procedure.



3.2 Propiedades del GridView para paginar

- Si desea que el GridView permita paginación debe asignar el valor True a la propiedad AllowPage
- La definición del tamaño de pagina lo define en la propiedad PageSize y si desea que la cabecera del GridView este siempre visible así no hayan registros que mostrar lo hace colocando en True la propiedad ShowHeaderWhenEmpty.
- Por ultimo el estilo de la paginación lo define en el grupo de propiedades PageStyle



3.3 El evento PageIndexChanging

- Para pasar de pagina en pagina en los registros del GridView debe programar el evento PageIndexChanging.
- A continuación mostramos el código del dicho evento para nuestro ejemplo del mantenimiento de proveedores:

```
protected void grvProveedor_PageIndexChanging(object sender, GridViewPageEventArgs e)
{
    grvProveedor.PageIndex = e.NewPageIndex;
    dtv = new DataView(objProveedorBL.ListarProveedor());
    Session["Vista"] = dtv;
    CargarDatos(txtFiltro.Text);
}
```

LABORATORIO

Implemente el mantenimiento de la tabla Tb_Proveedor de la base de datos VentasLeon. Tome como referencia el diseño del mantenimiento hecho en Windows para dicha tabla, y oriéntelo a WEB en función de lo aprendido con el empleo de controles de la ToolKit AJAX.



Conclusiones de la sesión:

- Las capas son reutilizables y nos permiten independizar la capa de presentación.
- Es importante interactuar con AJAX (u otra herramienta) que consolide una interface amigable para el mantenimiento de una tabla.
- Es importante paginar para poder mostrar de manera amigable y en secuencia de paginas el listado de los registros.

