When using an activeX control in Microsoft Internet Explorer you may need to catch and handle various events of vdraw yourself. This can be done through a script language. You can implement this using any script language out there, in this help file both a vbscript and a javascript example are demonstrated.
To use this code you can add it in an .aspx file and publish it in your server. You should always keep in mind that this code, since it's a browser script language, will be executed in the client's side. If you need data sent to your server, there are many web programming technologies you could use.

Javascript is one of the most popular script languages. Even though, handling ActiveX control events with javascript is not too complicated, vbscript handles them a lot easier. Javascript has other benefits too, so it's up to you to select what language best suits your needs.

If you are not sure how to use vdraw as an ActiveX control or you need more information on how to, check the following links.
http://www.vdraw.com/support/using-vdf-in-a-web-service/
http://www.vdraw.com/articles/howto/60000100-use-the-vdf-component-6005-and-above-in-html-%28internetexplorer%29/
http://www.vdraw.com/support/creating-a-web-installation-cab/

Sample code in Javascript:

```
<script language="javascript">
   //when creating an event handler you don't need to worry about the type of parameters. You only need to
   //have the correct number of parameters.
   function onAfterAddEntity(entity) {
      document.getElementById("log").value += "An entity was added with handle: " + entity.handle + "\n";
      document.getElementById("log").scrollTop = document.getElementById("log").scrollHeight;
   }
   function onAfterOpenDocument() {
      document.getElementById("log").value += "Opened " + myControl.activedocument.filename + "\n";
      document.getElementById("log").scrollTop = document.getElementById("log").scrollHeight;
   }
   function onMouseMove(x, y) {
      document.getElementById("XY").innerHTML = "X:" + x + " Y:" + y;
   }
   function onMouseDown(button, shift, x, y) {
      if (button != 1) return;
      var log = document.getElementById("log");
      log.value += "Clicked at: " + x + y + "\n";
      log.scrollTop = log.scrollHeight;
      //createinstance method is a way to overcome the limitations of javascript to recognize specific types.
      //Using createinstance you can get a number of different class objects, in this case we get a gPoint object.
      var pt = myControl.createinstance(23);
      //Now that we know our object is a gPoint, we can use all its methods and properties.
      pt.setvalueex(x, y, 0);
      //In order to get the real gPoint content of the variable and not the javascript object, we need to use the
      //value property of our object.
      var figure = myControl.activedocument.GetEntityFromPOINT(pt.value);
      if (figure == undefined) return;
      //Using the GetEntityFromPOINT method we managed to get the object located at where the user clicked. Now that
      //we have the figure, we can do whatever we want. Edit it, read properties, show the grips and generally, whatever
      //we could do if we were using vdraw locally.
      figure.showgrips = true;
      figure.invalidate();
      log.value += figure.handle + " " + figure.type + " object selected\n";
      log.scrollTop = log.scrollHeight;
   }
   //initvd is called to assign the vdraw activeX to the myControl variable.
   var myControl;
   function initvd() {
      myControl = document.getElementById("DVdraw1");
      //After calling the getElementById method of document, we check if myControl is undefined,
      //just in case something went wrong.
      if (myControl == undefined) {
         return;
      }
      //By default freezeevents and freezeentityevents are true, so no events will fire if not set
      //to false.
      myControl.freezeevents(false);
      myControl.freezeentityevents(false);
      //Using javascript, you can assign an eventhandler to an event by using attachEvent. The first
      //string is the event of our activeX control and the second the name of the javascript eventhandler.
      myControl.attachEvent('afteraddentity', onAfterAddEntity);
      myControl.attachEvent('afteropendocument', onAfterOpenDocument);
      myControl.attachEvent('CoordMouseMove', onMouseMove);
      myControl.attachEvent('MouseDown', onMouseDown);
   }
   function openFile() {
      myControl.activedocument.open("");
   }
   function makeLine() {
```

```
        myControl.CommandAction.cmdline("user");
    }
    function ClearSelection() {
        var entities = myControl.activedocument.model.entities;
        for (var i = 0; i < entities.count; i++) {
            var figure = entities(i);
            figure.showgrips = false;
        }
        myControl.redraw();
    }
</script>

<html>
<head>
    <style type="text/css">
        #log
        {
            width: 447px;
        }
        .style1
        {
            width: 439px;
        }
    </style>
</head>
<!-- initvd is called when the onload event of body is fired. When this event is fired, it means that all components
have finished loading. -->
<body onload="javascript:initvd();">
    <form id="form1">
    <table>
        <tr>
            <td>
                <object id="DVdraw1" style="width: 950px; height: 600px" codebase="http://www.vdraw.com/tnet/1/vdf.cab#version=-1,-1,-1,-1"
                    classid="clsid:50585480-E1F3-432A-A9D6-C694EDC9A12D" viewastext>
                    <param name="LicVAL" value="PUT THE SERVER'S LICVAL LICENSE HERE" />
                    <param name="PickSize" value="10" />
                </object>
            </td>
            <td class="style1">
                <textarea id="log" rows="20" wrap="off"></textarea>
                <label id="XY">
                    X:0 Y:0</label>
            </td>
        </tr>
    </table>
    <!-- adding event handlers to html components is real easy, as shown below.-->
    <button onclick="javascript:openFile();">
        Open File</button>
    <button onclick="javascript:javascript:makeLine();">
        Draw Line</button>
    <button onclick="javascript:ClearSelection();">
        Clear Selection</button>
    </form>
</body>
</html>
```

Sample code in Vbscript:

```
<script runat="server">

    Protected Sub Page_Load(ByVal sender As Object, ByVal e As System.EventArgs)

    End Sub
</script>

<script type="text/vbscript">
'Using vbscript you'll find that it's a lot easier to handle the activeX control's events. You just need to create
'an event handler using as name the activeX component's Id followed by an underscore "_" and the event's name. The
'event's name as it is in the activeX component.
sub DVdraw1_afteraddentity(entity)
        str = form2.log.value
        form2.log.value = str & "An entity was added with handle: " & entity.handle & vbCr &vblf
        form2.log.scrollTop = form2.log.scrollHeight
End sub
sub DVdraw1_AfterOpenDocument()
        str = form2.log.value
        form2.log.value = str & "Opened " + form2.DVdraw1.activedocument.filename & vbCr &vblf
        form2.log.scrollTop = form2.log.scrollHeight
End sub
sub DVdraw1_MouseMove(button, shift, x, y)
        set label = document.getElementById("xy")
        label.innerHTML = "X:" & x & " Y:" & y
End sub
sub DVdraw1_MouseDown(button, shift, x, y)
```

```
        if button <> 1 then exit sub
        dim str
        str = ""
        str = form2.log.value
        form2.log.value = str & "Clicked at: " & x & " " & y & vbCr &vblf
        form2.log.scrollTop = form2.log.scrollHeight
        'createinstance method is a way to overcome the limitations of javascript to recognize specific types.
        'Using createinstance you can get a number of different class objects, in this case we get a gPoint object.
        set pt = form2.DVdraw1.createinstance(23)
        'Now that we know our object is a gPoint, we can use all its methods and properties.
        pt.setvalueex x, y, 0
        'In order to get the real gPoint content of the variable and not the javascript object, we need to use the
        'value property of our object.
        set figure = form2.DVdraw1.activedocument.GetEntityFromPOINT(pt.value)
        if figure is nothing then exit sub
        'Using the GetEntityFromPOINT method we managed to get the object located at where the user clicked. Now that
        'we have the figure, we can do whatever we want. Edit it, read properties, show the grips and generally, whatever
        'we could do if we were using vdraw locally.
        figure.showgrips = true
        figure.invalidate
        str = form2.log.value
        form2.log.value = str & figure.handle & " " & figure.type & " object selected" & vbCr &vblf
        form2.log.scrollTop = form2.log.scrollHeight
End sub
    'initvd is called to assign the vdraw activeX to the myControl variable only in javascript. Using vbscript are quite
    'more simple. You don't need to have a global variable in order to use the control.
sub initvd()
        form2.DVdraw1.FreezeEvents false
        form2.DVdraw1.freezeentityevents false
        form2.log.value = ""
End sub
sub openFile()
        form2.DVdraw1.activedocument.open ""
End sub
sub makeLine()
        form2.DVdraw1.CommandAction.cmdline "user"
End sub
sub clearSelection()
        set entities = form2.DVdraw1.activedocument.model.entities
        for  i = 0 to entities.count-1
            set figure = entities(i)
            figure.showgrips = false
        Next
        form2.DVdraw1.redraw
End sub
</script>

<html>
<head>
    <style type="text/css">
        #log
        {
            width: 447px;
        }
        .style1
        {
            width: 439px;
        }
    </style>
</head>
<!-- initvd is called when the onload event of body is fired. When this event is fired, it means that all components
have finished loading. -->
<body id="body" onload="vbscript:initvd">
    <form id="form2">
    <table>
        <tr>
            <td>
                <object id="DVdraw1" style="width: 950px; height: 600px" codebase="http://www.vdraw.com/tnet/1/vdf.cab#version=-1,-1,-1,-1"
                    classid="clsid:50585480-E1F3-432A-A9D6-C694EDC9A12D" viewastext>
                    <param name="LicVAL" value="PUT THE SERVER'S LICVAL LICENSE HERE" />
                    <param name="PickSize" value="10" />
                </object>
            </td>
            <td class="style1">
                <textarea id="log" rows="20" wrap="off"></textarea>
                <label id="xy">
                    X:0 Y:0</label>
            </td>
        </tr>
    </table>
    <!-- adding event handlers to html components is real easy, as shown below.-->

    <button id="Button1" onclick="vbscript:openFile">
        Open File</button>
    <button id="Button2" onclick="vbscript:makeLine">
```

```
        Draw Line</button>
    <button id="Button3" onclick="vbscript:clearSelection">
        Clear Selection</button>
    </form>
</body>
</html>
```