

# LiveConnect: руководство для IE

## Walk

2 ways to call JS functions from applets:

1. Netscape's LiveConnect
2. JavaBeans' outgoing events

The 1st way should work in both IE and NS and the JavaPlugin from Sun, the second only works in IE since it relies on JavaCOM.

1. Use the `call()` or `eval()` methods in `netscape.javascript.JSObject`, which is documented in the LiveConnect section on the NS manuals on JS: [developer.netscape.com/docs/manuals/javascript.html](http://developer.netscape.com/docs/manuals/javascript.html) (the relevant pages are attached)
2. This is tricky and almost undocumented and works this way: a Java class works like an ActiveX control in IE and IE knows how to manage it relying on JavaBean design patterns (with a few modifications), that's to say:
  - only public methods are used and no polymorphism is supported
  - to access properties you must supply `getX()` and/or `setX()` methods (`getX()` for readable props, `setX()` for writable ones, both for R/W props) (thus you can either use the access methods or the property from JS)
  - values are converted on the fly: JS strings to `String`, JS numbers to `int`, `float`... or whatever, JS objects to any Java object indicated, Java objects are wrapped in a COM Variant to old their dispatch interface
  - the bean class can generate events implementing a pair of methods `addXListener(XListener)` and `removeXListener(XListener)`; you have to define an interface `XListener` (of course you can use the ones in `java.awt.event.*` if they suits your need) which extends `java.util.EventListener`; the abstract methods in the interface should return `void` and accept an `XEvent` as parameter (which extends `java.util.EventObject`) Once you have setup those 3 beans, you use them this way: v embed the applet (well, it needs not be an Applet actually, it can be any

java.awt.Component — if it must be rendered graphically — or any class if no display is needed) using the <OBJECT> tag and the "java:" moniker:

```
<OBJECT ID="xappid"  
CLASSID="java:mypkg.XApplet"></OBJECT>
```

- set the CSS display property to "hidden" if no visual rendering is needed and the WIDTH and HEIGHT attributes/properties; remember to give it an ID if you want to easily access the bean from JS; remember to put XApplet.class in the classpath of the browser or in CODEBASE and use the full name if it's in a package other than the empty one (and in such case take into account any needed directory tree: that is, create a mypkg dir and put the .class there); know that IE uses "." as default CODEBASE

- instruct IE to setup an XListener:

```
<SCRIPT LANGUAGE="JScript"  
FOR="xappid" EVENT="someXEvent(e)">  
  
// function code here: global code, not function() {}  
// the var "e" is an XEvent object  
  
</SCRIPT>
```

you do not have to create an event handler for each method in XListener, only the one you need: IE will implement stub methods for the others

- raise events in you bean whenever you need:

```
synchronized(xlisteners) {  
    for(int i = 0; i < xlisteners.size(); i++) {  
        XEvent e = new XEvent();  
        ((XListener)xlisteners.elementAt(i)).someXEvent(e);  
    }  
}
```

this assumes that you have somewhere

```
private Vector xlisteners = new Vector();  
  
public void addXListener(XListener xl) {  
    synchronized(xlisteners) {  
        xlisteners.addElement(xl);  
    }  
}
```

## *LiveConnect: руководство для IE*

```
    }  
}  
  
public void removeXListener(XListener xl) {  
    synchronized(xlisteners) {  
        xlisteners.removeElement(xl);  
    }  
}
```

And this is the whole story. Most likely you need not the complexites (but the power!) of JavaCOM in IE and JScript is enough.