
InetSupervisor- FlashSDK_UsersGuide

Copyright © 2005 Quark Communications Inc. All rights reserved.
Revised - August 4, 2008

1.0 Flash Fundamentals

1.1 Flash

This guide makes the assumption that while the user may be new to **Macromedia Flash**, he/she has a basic knowledge of C type computer coding and scripting. **Quark Communications Inc.** recommends that all users obtain and read Macromedia's user guides and help files for Flash.

1.2 Communicator

All communications between client and server is now handled by one flash component called the communicator (*Ex-Communicator.swf*). Since the flash components communicate only to the communicator one must be present on the web site at all times in order to receive point value information and to send overrides. The communicator can be but does not need to be housed on the same page as the receiving flash but it must be in a currently opened page. For example the default web site contains the communicator in the left side navigation pane, not in the center information pane where all the flash is located. Due to the fact that the navigation pane does not close, the communicator is always on and transferring data to and from any flash that is currently running. The communicator also submits overrides and will check for the proper user level before sending the overriding value (*please see the Inetsupervisor user guide for user level information*).

1.3 Connector Component

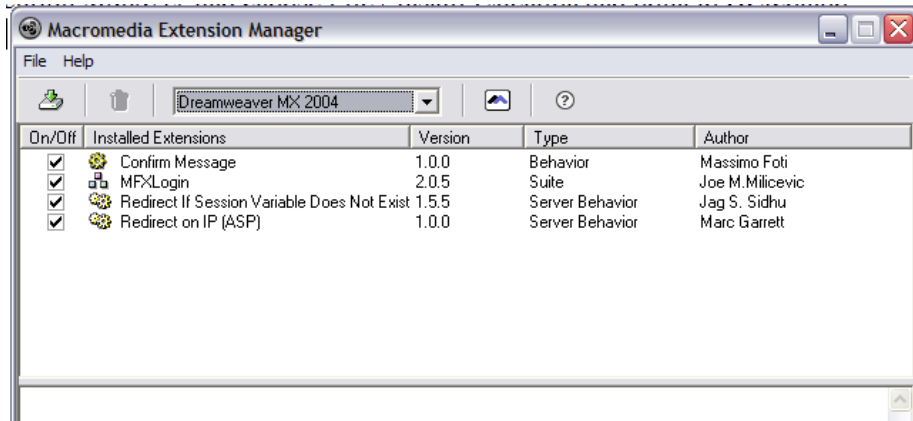
The connector is a flash component, made by Quark Communications Inc., to enable it's customers to make there own flash movies that communicate with the Inetsupervisor database. The connector handles all of the connection/communication with the Inetsupervisor database to simplify and ease the making of interactive user graphics. Note Quark Communications Inc. recomends that it's connector be placed in it's own layer in the flash movie, and that that layer spans the length of all the frames in the movie.

2.0 Installing and using QCI's Connection Component

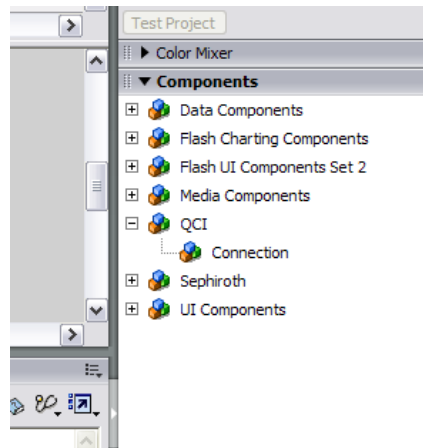
2.1 Installing the Connection Component

To transfer information to and from the communicator, it is necessary to attach a connection component to every flash movie created. The connection component has been made into a self installing file called *QCIcomponents.mxp*. To start using the connection component first install *Macromedia Flash MX2004* or later then:

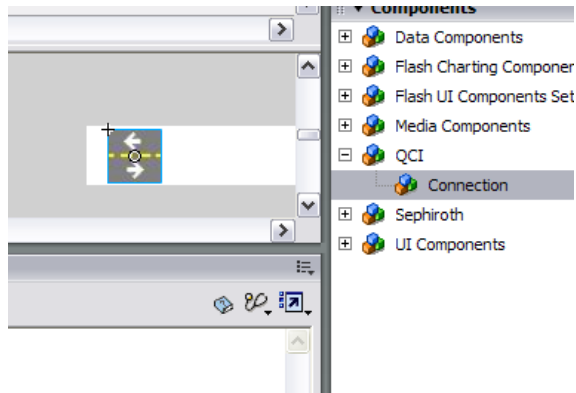
1. Install the QCIcomponents.mxp file in a temporary folder and double click or open the **Macromedia Extension Manager** and choose File / Install Extension and point to QCI-components.mxp.



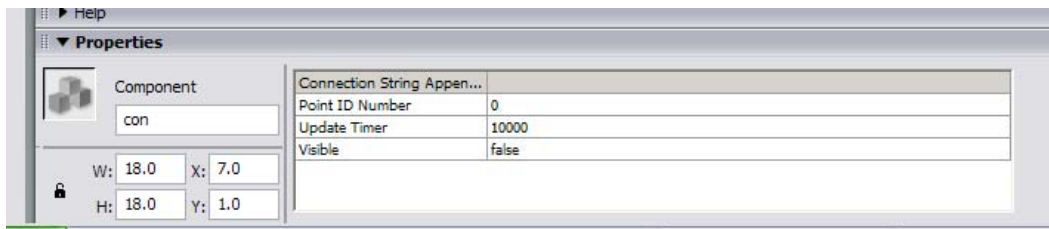
2. Read and accept the disclaimer
3. When the component has finished installing click O.K. and close manager
4. Open Macromedia Flash
5. Create a new flash movie
6. Open the components panel from window / development panels and check mark component panel or use Ctrl+F7



- Expand the QCI then drag and drop the connection component onto your movie



- Give your connection an instance name in the properties box.



2.2 Parameters of the Connection Component

In the properties area you will find 4 adjustable parameters for the connection - Connection String Appendage, Point ID Number, Update Timer, and Visible.

Connection String Appendage

The connection string is a unique identification created dynamically during movie load time and used by the communicator to identify every flash movie individually so each flash movie receives it's proper values. The connection component automatically takes care of creating and sending this connection string to the communicator. If necessary the connection string can be appended with this parameter.

Point ID Number

This is the point id number created in the points table and used to identify individual points in the database. The point id may be “hard coded” in this parameter but more often you will wish to make it user definable which we will see later in this guide.

Update Timer

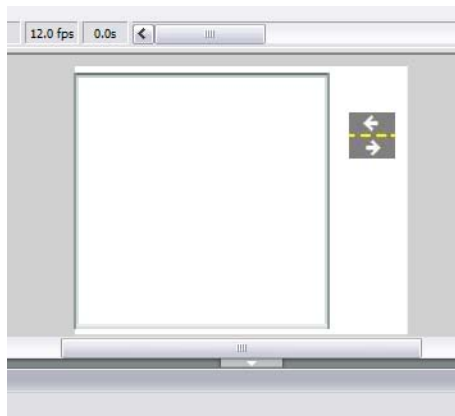
This is the time delay, in milliseconds, between calls to the communicator for a point value update. The communicator is called immediately during movie load and then subsequent calls are made after the time delay has passed. The default is 10 seconds.

Visible

This parameter is used to determine if the component is visible or invisible.

2.3 Using the connection component to display values

1. Open a new flash document and drag and drop the connection component onto the flash movie. Give it the instance name of *con*. Create a new layer then Drag and drop a TextArea component with the size of 100 pixels wide and 100 pixels tall onto the flash movie also, with the instance name of *status_txt*.

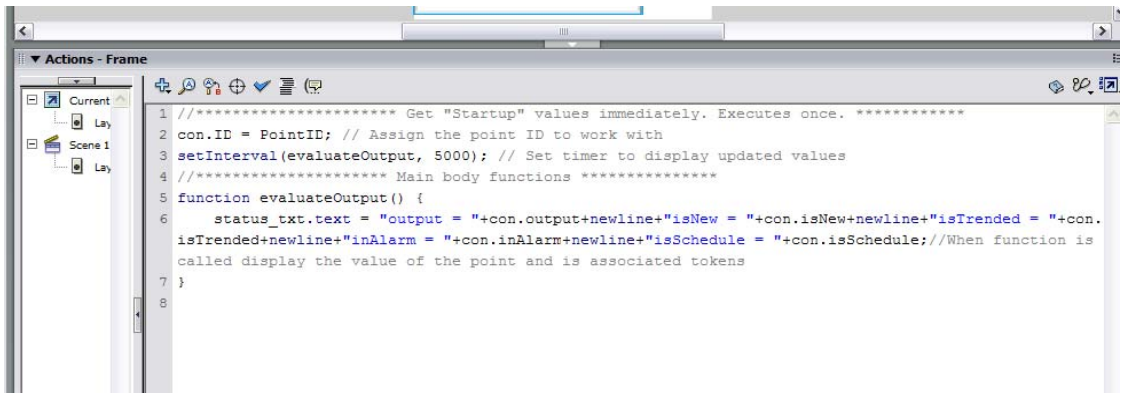


2. Highlight frame 1 of the text box layer and place the following action script into the Actions panel

```
con.ID = PointID;
```

```
setInterval(evaluateOutput, 5000);
```

```
function evaluateOutput() {status_txt.text = "output = "+con.output+newline+"isNew = "+con.isNew+newline+"isTrended = "+con.isTrended+newline+"inAlarm = "+con.inAlarm+newline+"isSchedule = "+con.isSchedule;}
```



3. Publish the flash movie to your desired website flash folder (Shift+F12).
4. Place the new .swf file onto a webpage with the communicator. Identify the Point ID number by adding the `<param>` tag with `"flashvars"` as the name and with `"PointID=XXX"` as the value, where `XXX` is the database point id number, please see Inetsupervisor user's guide for a further description of this process. Publish the webpage.

```
output = ON
isNew = 1
isTrended = 1
inAlarm = 0
isSchedule = 0
```

The script illustrates how to make the Point ID user defined and how to receive information from the connection.

1. Line 1 assignees the value of PointID, given to us by the **PointID= "flashvars"** param tag, the variable ID associated with the component named con.

```
con.ID = PointID;
```

2. Line 2 creates a timer that calls the function evaluateOutput every 5 seconds

```
setInterval(evaluateOutput, 5000);
```

3. Line 3 is the function evaluateOutput. When the function is called it assignees the value of the point id given(con.output) and the value of it's associated tokens (con.isNew, con.isTrended, con.inAlarm, con.isSchedule) to the text of status_txt.

```
function evaluateOutput() {status_txt.text = "output = "+con.output+newline+"isNew = "+con.isNew+newline+"isTrended = "+con.isTrended+newline+"inAlarm = "+con.inAlarm+newline+"isSchedule = "+con.isSchedule;}
```

The tokens are used for passing additional information about the database point beyond just the value.

instanceName.output

This is the point value as it is current in the SQL database. This value may be anything from a number to a text string. If the point is digital the only values passed from the communicator are the strings "ON" and "OFF".

instanceName.isNew

The value of 1 will be sent if the point update delivered time is newer than the current server time minus the value (in seconds) set in the web.config file. If not then the value of 0 will be sent.

instanceName.isTrended

The value of 1 will be sent if the point's isTrendable check box is checked. If not then the value of 0 will be sent.

instanceName.inAlarm

The value of 1 will be sent if the point is currently in alarm. If not then the value of 0 will be sent.

instanceName.isSchedule

The value of 1 will be sent if the point's isSchedule check box is checked. If not then the value of 0 will be sent.

instanceName.ID

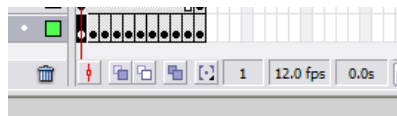
The database identification number of the point.

3.0 Controlling Graphics

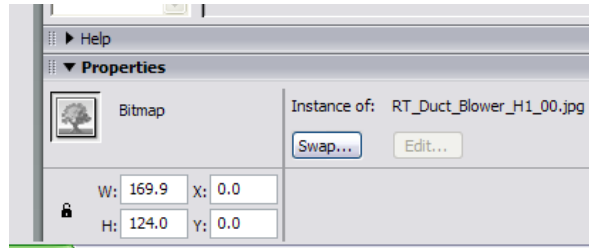
This chapter will show how to use the point value to control 2 kinds of graphics, continuous motion like a fan and position and stop like a valve.

3.1 Controlling a Fan

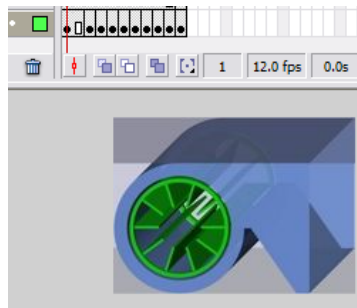
1. Open a new flash document with a 170 by 124 pixel width and height.
2. Go to File/ Import/ Import to Library and import the 10 fan .jpegs with this SDK.
3. Go to the timeline of layer 1, right click and add 10 more keyframes.



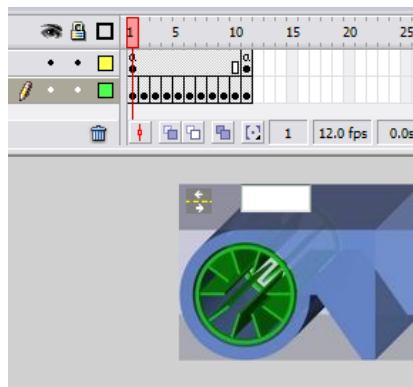
4. Drag and drop the first fan picture on the first and second frames and position them exactly with the background using the properties panel.



5. Do the same thing with the next frames using the next corresponding pictures.



6. Insert a new layer. Then drag and drop the connection component and a TextArea component onto to the movie. You may place these component's any place on the movie. Associate these with the first frame of the 2nd layer. Name the TextArea *status_txt* and size it to 50 by 20 pixel width and hieght. Name the connection *con*.



7. Create a new layer.
8. Make the last frame of the third layer a keyframe.
9. Insert the following code into the action panel of the first frame in the third layer.

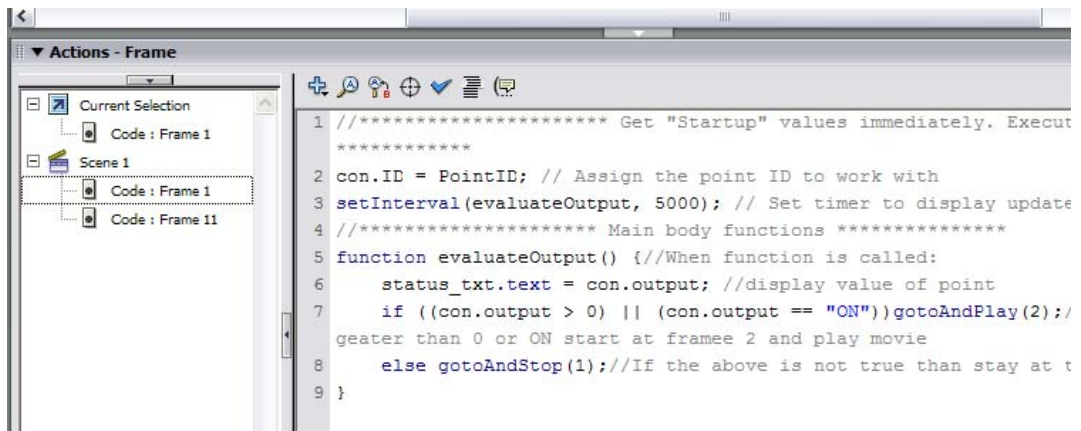
```
con.ID = PointID;
```

```
setInterval(evaluateOutput, 5000);
```

```
function evaluateOutput() {status_txt.text = con.output;
```

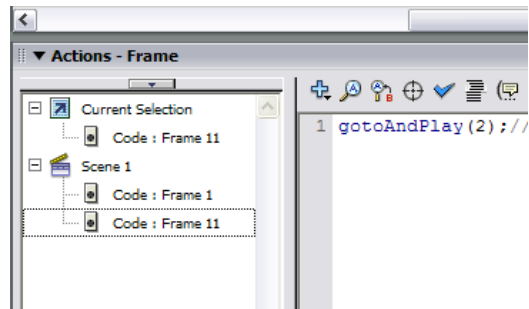
```
if((con.output > 0) || (con.output == "ON"))gotoAndPlay(2);
```

```
else gotoAndStop(1);}
```



10. Insert the following code into the action panel of the last frame in the third layer.

gotoAndPlay(2);



11. Publish the flash movie to your desired website flash folder (Shift+F12).
12. Place the new .swf file onto a webpage with the communicator. Identify the Point ID number by adding the *<param>* tag with “*flashvars*” as the name and with “*PointID=XXX*” as the value, where *XXX* is the database point id number, please see Inetsupervisor user’s guide for a further description of this process. Publish the webpage.

The script illustrates how to use the point value to control when to start running a movie.

1. Line 1 assignees the value of PointID, given to us by the *PointID= “flashvars”* param tag, the variable ID associated with the component named con.

con.ID = PointID;

2. Line 2 creates a timer that calls the function evaluateOutput every 5 seconds

setInterval(evaluateOutput, 5000);

3. Line 3 is the function evaluateOutput. When the function is called it assignees the value of the point id given(con.output) to the text of status_txt. Next it looks at the value of the point and if the value is greater than 0 or a string with the text “ON” the movie will move to frame 2 and start running. If the value is not either of those cases then it stays at frame one and stops.

function evaluateOutput() {status_txt.text = con.output;

if ((con.output > 0) || (con.output == "ON"))gotoAndPlay(2);

else gotoAndStop(1);}

-
4. The last frame script shows that if the movie is running and goes to the end frame then go back to frame 2 and start all over again to loop the movie action.

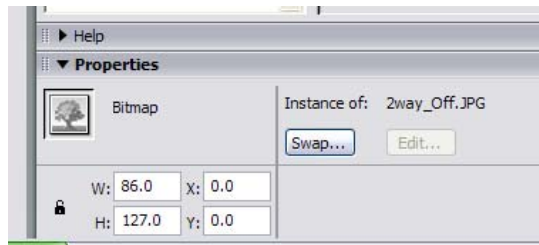
gotoAndPlay(2);

3.2 Controlling a Valve

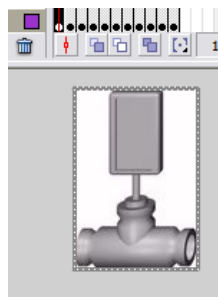
1. Open a new flash document with a 86 by 127 pixel width and height.
2. Go to File/ Import/ Import to Library and import the 10 valve .jpgs with this SDK.
3. Go to the timeline of layer 1, right click and add 10 more keyframes.



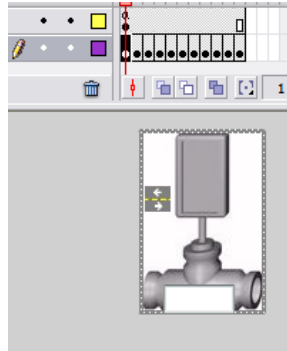
4. Drag and drop the first valve picture on the first frame and position them exactly with the background using the properties panel.



5. Do the same thing with the next frames using the next corresponding pictures.



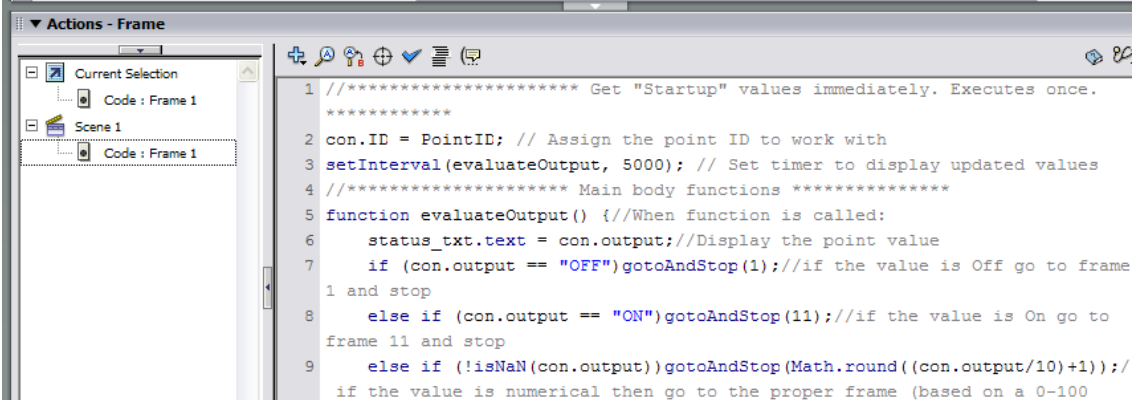
-
6. Insert a new layer. Then drag and drop the connection component and a TextArea component onto to the movie. You may place these component's any place on the movie. Associate these with the first frame of the 2nd layer. Name the TextArea *status_txt* and size it to 50 by 20 pixel width and hieght. Name the connection *con*.



7. Create a new layer.
8. Insert the following code into the action panel of the first frame in the third layer.

```
con.ID = PointID;  
  
setInterval(evaluateOutput, 5000);  
  
function evaluateOutput() {status_txt.text = con.output;  
  
if (con.output == "OFF")gotoAndStop(1);  
  
else if (con.output == "ON")gotoAndStop(11);
```

else if (!isNaN(con.output))gotoAndStop(Math.round((con.output/10)+1));}



```
1 //***** Get "Startup" values immediately. Executes once.
  *****
2 con.ID = PointID; // Assign the point ID to work with
3 setInterval(evaluateOutput, 5000); // Set timer to display updated values
4 //***** Main body functions *****
5 function evaluateOutput() { //When function is called:
6     status_txt.text = con.output; //Display the point value
7     if (con.output == "OFF") gotoAndStop(1); //if the value is Off go to frame
  1 and stop
8     else if (con.output == "ON") gotoAndStop(11); //if the value is On go to
  frame 11 and stop
9     else if (!isNaN(con.output)) gotoAndStop(Math.round((con.output/10)+1)); //
  if the value is numerical then go to the proper frame (based on a 0-100
```

9. Publish the flash movie to your desired website flash folder (Shift+F12).
10. Place the new .swf file onto a webpage with the communicator. Identify the Point ID number by adding the `<param>` tag with *“flashvars”* as the name and with *“PointID=XXX”* as the value, where *XXX* is the database point id number, please see Inetsupervisor user’s guide for a further description of this process. Publish the webpage.

The script illustrates how to use the point value to control what frame of a movie to goto and hold.

1. Line 1 assignes the value of PointID, given to us by the *PointID= “flashvars”* param tag, the variable ID associated with the component named con.

```
con.ID = PointID;
```

2. Line 2 creates a timer that calls the function evaluateOutput every 5 seconds

```
setInterval(evaluateOutput, 5000);
```

3. Line 3 is the function evaluateOutput. When the function is called it assignes the value of the point id given(con.output) to the text of status_txt. Next it looks at the value of the point and if the value is greater is a string with the text “OFF” the movie will stay to frame 1. If the string has the text “ON” the movie will got frame 11 and stay. If the value is a number then the movie will goto the frame closest to the value of the point divided by 10 plus 1 frame.

```
function evaluateOutput() {status_txt.text = con.output;
```

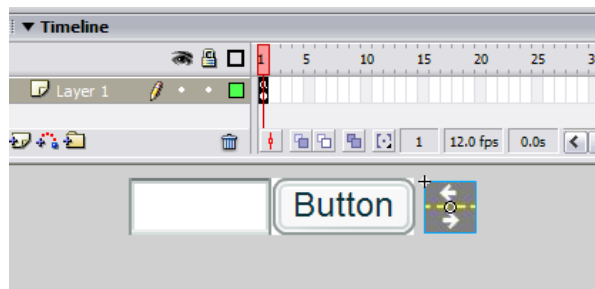
```
if (con.output == "OFF")gotoAndStop(1);
```

```
else if (con.output == "ON")gotoAndStop(11);
```

```
else if (!isNaN(con.output))gotoAndStop(Math.round((con.output/10)+1));}
```

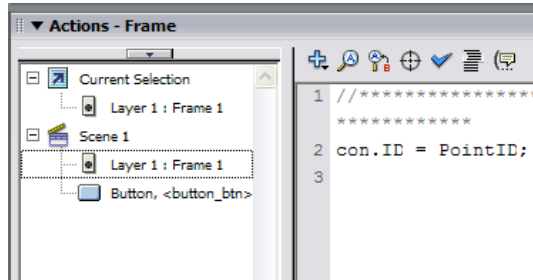
4.0 Sending an Override

1. Open a new flash document with a 100 by 20 pixel width and height.
2. Drag and drop the a TextArea component, and a Button component onto to the movie. Place the TextArea component to the left side of the movie and the button just to it's right. Name the TextInput *input_txt* and size it to 50 by 20 pixel width and hieght. Name the button *button_btn* and size it to 50 by 20 pixels width and hieght.
3. Create a new layer and drag and drop the connection component, Name the connection *con*. Note the connection component does not need to be placed on the movie back ground to work.



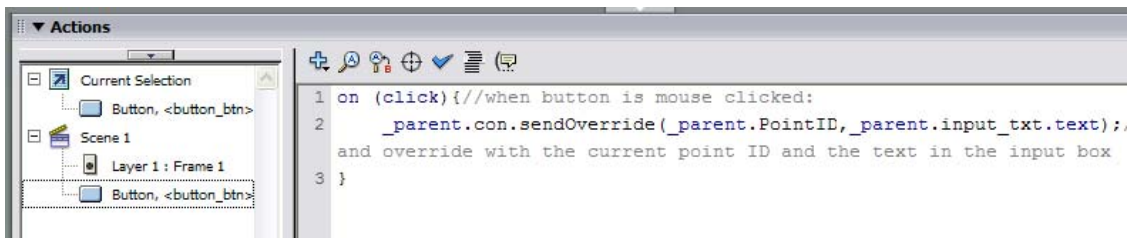
4. Insert the following code into the action panel of the first frame.

con.ID = PointID;



5. Insert the following code in to the action panel of the button

on (click){_parent.con.sendOverride(_parent.PointID,_parent.input_txt.text);}



6. Publish the flash movie to your desired website flash folder (Shift+F12).
7. Place the new .swf file onto a webpage with the communicator. Identify the Point ID number by adding the *<param>* tag with “*flashvars*” as the name and with “*PointID=XXX*” as the value, where *XXX* is the database point id number, please see Inetsupervisor user’s guide for a further description of this process. Publish the webpage.

The script illustrates how to use build in *sendOverride* function in the connection component to send a value to the server for overrides.

1. Line 1 assignes the value of PointID, given to us by the *PointID= “flashvars”* param tag, the variable ID associated with the component named con.

con.ID = PointID;

2. Line 2 calls the *sendOverride* function of the component and feeds first the Point ID to be overridden then the value to override with, in this case the text typed in the TextInput.

```
on (click){_parent.con.sendOverride(_parent.PointID,_parent.input_txt.text);}
```

5.0 Url

If you are using the communicator frame-work provided with the CD then the communicator can also send the site URL information to the connector. To receive this information assign a string variable to the connector name.url (i.e *var myVar:String = connectorName.url*). The URL information name will be a string with following text *http://**domain name/root folder***(if used)/.

Visit us at **www.inetsupervisor.com**

Technical Support Phone: +1 (760) 634 6845

Technical Support e-mail: support@quarkcommunications.us

Sales e-mail: sales@quarkcommunications.us

