

# Basic Scripts

## Script Anatomy

All scripts are added to a frame in the same layer:

This code starts the playback head moving forward:

```
play_btn.addEventListener(MouseEvent.CLICK, playAnimation);  
  
function playAnimation(evt:MouseEvent):void {  
    play();  
}
```

This code stops the playback head from moving forward:

```
stop_btn.addEventListener(MouseEvent.CLICK, stopAnimation);  
function stopAnimation(evt:MouseEvent):void {  
    stop();  
}
```

### BreakDown...

#### Instance Target

```
play_btn  
stop_btn
```

#### Event Listener Type

```
MouseEvent
```

#### What is Being Listened For

```
CLICK
```

#### Function Being called

```
playAnimation  
stopAnimation
```

This is what a script added directly to a frame looks like

```
//action goes here
```

example:

```
play();
```

## Event (for targeted buttons / movieclips)

You have to add an EventListener:

```
addEventListener();
```

You have to target the 'button' or 'movieclip' that is going to trigger the event:

```
button_btn.addEventListener();
```

**or**

```
pressme_mc.addEventListener();
```

You have to add the following information within the Event Listener's Parentheses:

```
button_btn.addEventListener(MouseEvent.CLICK);
```

**or**

```
pressme_mc.addEventListener(MouseEvent.CLICK);
```

### A choice of MouseEvents are as follows:

```
MOUSE_UP  
MOUSE_DOWN  
CLICK  
DOUBLE_CLICK  
MOUSE_MOVE  
MOUSE_OVER  
MOUSE_OUT  
ROLL_OVER  
ROLL_OUT  
MOUSE_WHEEL
```

Add a comma after the MouseEvent and event choice, then identify the function to call:

```
button_btn.addEventListener(MouseEvent.CLICK, playAnimation);
```

**or**

```
pressme_mc.addEventListener(MouseEvent.CLICK, rotateMe);
```

Create the function:

```
function playAnimation(){  
}
```

Add the evt parameters inside the parenthesis and type in MouseEvent first identified within the EventListener:

```
function playAnimation(evt:MouseEvent){  
}
```

Add a colon and void after the parenthesis (void tells the function not to return any values):

```
function playAnimation(evt:MouseEvent):void{  
}
```

Define the action:

```
function playAnimation(evt:MouseEvent):void{  
    play();  
}
```

**or**

```
function rotateMe(evt:MouseEvent):void{  
    rotate_mc.rotation+=10;  
}
```

**All together now:**

```
button_btn.addEventListener(MouseEvent.CLICK, playAnimation);  
function playAnimation(evt:MouseEvent):void{  
    play();  
}
```

**or**

```
pressme_mc.addEventListener(MouseEvent.CLICK, rotateMe);  
function rotateMe(evt:MouseEvent):void{  
    rotate_mc.rotation+=10;  
}
```

### Action (all actions are contained within functions!)

Below are actions; whether you add the event or no event depends on where the action is placed! If the action directly affects the frame (for example) you do not need to contain it in a function for it to work:

Do not add anything inside parenthesis (these actions directly affect the playback head when it enters the frame where this action is typed:

```
play();  
stop();
```

Add frame number or frame label inside parenthesis

```
gotoAndPlay();  
gotoAndStop();
```

example:

```
gotoAndPlay(3);  
gotoAndPlay("home");
```

### Drag a targeted movieclip

evt refers to the targeted movieclip identified within the EventListener:

```
evt.target.startDrag();
```

stop dragging all movieclips

```
evt.target.stopDrag();
```

example:

```
myMovieClip_mc.addEventListener(MouseEvent.CLICK, dragMe);  
myMovieClip_mc.addEventListener(MouseEvent.CLICK, stopDragMe);  
function dragMe (evt:MouseEvent):void{  
    evt.target.startDrag();  
}  
function stopDragMe (evt:MouseEvent):void{  
    evt.target.stopDrag();  
}
```

## Change the property of a movieclip

Refer first to movieclip you want to target followed by . and the property you want to affect followed by = then the value you want to change:

```
instancename.alpha = .2;
```

alpha	//(0 to 1) percentage
x	//refers to x coordinate on stage
y	//refers to y coordinate on stage
scaleX	//(0-1)percentage: default is 1
scaleY	//(0-1)percentage: default is 1
rotation	//degree: 0 - 360
mouseX	//refers to x coordinate of mouse cursor on stage
mouseY	//refers to y coordinate of mouse cursor on stage
visible	//default is 1 (true); turn off object is 0 (false)

## Check Collision between 2 movieclips (hitTestObject)

Use with draggable object; replace instancename with name of each movieclip:

```
if (evt.target.hitTestObject(big_mc)) {  
    //action here  
}
```

example:

```
small_mc.addEventListener(MouseEvent.CLICK, dragMe);  
small_mc.addEventListener(MouseEvent.CLICK, stopDragMe);  
small_mc.addEventListener(MouseEvent.CLICK, scaleMe);  
  
function dragMe(evt:MouseEvent):void {  
    evt.target.startDrag(true);  
}  
  
function stopDragMe(evt:MouseEvent):void {  
    evt.target.stopDrag();  
}  
  
function scaleMe(evt:MouseEvent):void {  
    if (evt.target.hitTestObject(big_mc)) {  
        evt.target.scaleX=.5;  
        evt.target.scaleY=.5;  
    }  
}
```

## **Comment**

Anything 'commented out' is ignored by program

Single line comment:

```
// your comments go here
```

Multi line comment:

```
/* your comments go here  
your comments go here  
your comments go here  
your comments go here */
```

## **Trace**

Displays text in output window. good for checking script to see if it works!

```
trace("outputwindowdisplayhere");
```