

Getting Started

Phan I cua cuon sach se ghi lai nhung cai nhin tong quan nhat. Chapter 1 va 2 se la nhung cai tong quat ve actionscript, dua ra nhung danh sach ve nhung net dac trung , tom tat giai nghia nhung dac trung cau lap trinh huong hoi tuong (OOP - object-oriented programming) va chu y quan trong nhat ve su huu dung cua quyen sach nay`.

ND Phan 1 trinh bay khai quat chung ve co so cua ngon ngu, no co the su dung cho moi phien van cua actionscript.

(Phan nay` dai` qua` va vo bo? Nen minh hok dich – thong cam L).

Chapter1: ACTIONSCRIPT OVERVIEW

I.L p – ây là n n t ng c a l p trình OOP r t ph bi n hi n nay. Mình không i sâu vào l p trình tuy n tính (mã l nh th c hi n theo 1 chi u t trên xu ng) hay l p trình c u trúc (s d ng hàm, th t c) vì nh ng cái này x a roài, nó c ng không liên n l p trình h ng i t ng-OOP. Chúng ta N/C v l p nh 1 VD kh i u, sau nay mình s d ch chi ti t v OOP chapter 6

Hay bat dau voi nhung vi du don gian ma ban se su dung chung trong thoi gian dai`, do la cau lenh **trace("NOI DUNG")**; nhung noi dung trong cau lenh se in ra o man hinh output cua flash va khong lam thay doi j i` chuong trinh`. Day la 1 cong cu sua loi rat hieu qua.

Bay gio hay mo notepad moi va go doan code sau vao

package {

```
import flash.display.MovieClip;
public class Main extends MovieClip {
    public function Main() {
        trace("Flash");
    }
}
```

Day la 1 doan code dung de khai bao 1 lop (class) trong flash co ten la main. Trong mot doan ma lenh. Co bao nhieu dau “{“ thi cung se co bay nhieu dau “}”

package {

}

Package co cau truc bat buoc. Can dam bao rang lop cua ban (class) co the bien dich dc.

import flash.display.MovieClip;

cau lenh dung de nhap class cua **MovieClip** (mot doi tuong san co trong flash) vao flash.

public class Main extends MovieClip

cau lenh dung de khai bao cho flash biet rang – lop main ma ta vua khai bao co the ke thua` nhung

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER 'S GUIDE J
thuoc tinh cua MovieClip san co trong flash. **Public** se lam cau lenh sau no duoc thuc hien ngay khi duoc load vao flash.

```
public function Main() {  
    trace("Flash");  
}
```

Day la buoc cua ban. Ban muon lop (class) lam ji` thi` hay go~ ma~ lenh vao doan giua cua code nay`. Tat nhien co the ban chua biet ji ve AS3 va do do chung ta se de trong phan nay`. Ban co the go vao **trace("Flash")**; day la 1 file class va ban co the save lai duoi ten tuy y song can co duoi la “.as” sau do luu no lai cung voi thu muc chua file flash sap duoc tao (sau nay ban se hoc cach luu file class o nhung thu muc khac). Luu y la len dat ten file class de nho' 1 ti' (nen dat la “Main.as”).

Bay gio hay tao 1 file flash, luu lai file nay duoi ten GettingStarted (tat nhien co duoi la “.fla”).

Dieu chinh nhu hinh` ve de file class co the load truc tiep vao flash Luu y de goi file Main.as ta chi viet vao Document class chu “Main” thay cho “Main.as”.



Nhan Ctrl + Enter va` xem dieu ji` xay ra :D.

LegacyCode Compatibility

Tinh ke thua AS3 duoc tiep noi tu AS2. tuy nhien 1 file swf khong the co Ma lenh cua AS2 va AS3 cung` ton tai. Mac du` flash CS3 van co the load dc 1 file flash viet bang AS2 nhung no khong the can thiep sua doi file nay. Mat khac flash CS3 co 2 che do lam viec – nhung ng` da quen AS2 va` hok muon thay doi va nhung ng` dung` AS3 se tan huong nhung hieu qua cua ngon ngu lai mang lai.

Ban nen hoc cach cap nhat moi lai AS3 dua tren ma lenh AS1 va AS2.

Neu flash 8 tro? Xuong' thi` khong the load file SWF co chua AS3.

CHAPTER 2 Core Language fundamentals

(nhung Co So co ban ve NNLT ActionScript)

1. Miscellaneous Basics (don gian pha tap ma lenh)

- 1 thuc hien theo trinh tu
- 2 tu` tren xuong duoi.
- 3 Tu` trai qua phai.

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER 'S GUIDE J

- 4 Su dung dau cham phay (;) khi ket thuc cau lenh. Dieu nay khong bat buoc nhung no se lam ro nghia cau lenh voi flash
- 5 Danh gia mot bieu thuc
- 6 VI DU: $x = x + 1;$ se gan' gia tri cua x bang $x + 1$ hay tang gia tri cua x len 1 don vi. (x co the khai bao theo kieu so').
- 7 Viec danh gia bieu thuc se duoc thuc hien bang phep gan dau “=” no khac voi toan tu so sanh bang “==” cua bieu thuc logic.
- 8 hay su dung cau lenh **trace** va ghi chu (dat // truoc nd ghi chu tren 1 dong) moi luc co the ban khong can dung khi lam nhung vi du nho, nhung voi 1 chuong trinh dai thi neu hom sau ban co chac se nho nhung ji minh lam dc ngay` hom qua? noi chung neu ban khong lam ji mo` am` thi lam on ghi chu lai cho ng` khac nho`. Hon nua viec su dung lenh trace se giup ban kiem soat tot chuong trinh dang viet.

2. Gia tri va kieu du lieu

VI DU: **var myVariable:Number = 1;**

De khai bao ta dung **var** tengiatri:kieuugiatri = gia tri;

Cong doan nay khong bat buoc, nghia la neu nhu ta viet **tengiatri = 1;** thi flash tu hieu tengiatri khai bao kieu so va gan 1 cho tengiatri.

Dat ten co the gom ca chu va so , it nhat cung can 1 ky tu va khong nen bat dau bang so hay 1 tu dung` rieng/lenh trong actionscript, khong dat 2 ten gia tri trung nhau

Duoit day la 1 so kieu du kieu trong AS:

Table 2-1. Variable types

Data type	Example	Description
Number	4.5	Any number, including floating point values (decimals)
int	-5	Any integer or whole number
uint	1	Unsigned integer or any non-negative whole number
String	"hello"	Text or a string of characters
Boolean	true	True or false
Array	[2, 9, 17]	More than one value in a single variable
Object	myObject	The basic structure of every ActionScript entity but also a custom form that can be used to store multiple values as an alternative to Array.

Number: kieu so co the la so thuc hoac nguyen.

Int: so nguyen hoac tong so.

String: chuoi ky tu co the dung **char** de khai bao 1 ky tu.

Object: kieu du lieu doi tuong, thuc the ton tai trong flash, dung cho nhung kieu du lieu phuc tap thay cho kieu mang du lieu (**ARRAY**).

Tren day la nhung kieu du lieu rat don gian, trong flash ta co the khai bao nhieu doi tuong trong flash bang ma lenh

VI DU: **var myMC:MovieClip = new MovieClip();**

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER 'S GUIDE J

Khai bao mot MovieClip moi voi ten MC

Trong nhieu phien ban truoc cua AS, phan khai bao nay khong bat buoc song voi AS3, yeu cau nay la bat buoc, cong viec nay co ve lam mat nhieu thoi gian song ban se thay hieu qua khi chay chuong trinh va sua loi va chung cung cap phan hoi ve dac trung cua chuong trinh.

3. Dieu kien

Bieu thuc dieu kien hay bieu thuc logic la mot dieu khong the thieu cua tat ca NNLT, AS3 cung vay, bieu thuc chi nhan 2 gia tri : **dung**' hoac **sai**.

Cau lenh **if** thi qua quen thuoc roi co le khong nen de cap nua vi cach dung giong voi AS2.

Cac toan tu gom >,<,>= ,<= ,><, == , && (and), || (or), ! (not).

VI DU:

```
var a:Number = 1;
var b:String = "hello";
var c:Boolean = false;
if (a == 1) {
    trace("option a");
}
// thong bao o output: option a
if (a == 1 && b == "goodbye") {
    trace("options a and b");
}
// cha co ji ca
if (a == 1 || b == "goodbye") {
    trace("option a or b");
}
// thong bao o output: option a or b
if (!c) {
    trace("not option c");
}
// thong bao o output: not option c
```

VI DU:

```
var a:Number = 1;
var b:String = "hello";
var c:Boolean = false;
if (a == 1) {
    trace("option a");
    // thong bao o output: option a
} else if (b == "hello") {
    trace("option b");
}
// thong bao o output: option b
```

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER 'S GUIDE

```
} else {  
    trace("option other");  
// cha co thong bao ji  
}
```

4. switch

Khi go qua nhieu lenh if se rat moi tay mo` mat, do do' switch la 1 gai phap thay the tot.

Vd:

```
switch (a) {  
case 1 :  
trace("one");  
break;  
case 2 :  
trace("two");  
break;  
case 3 :  
trace("three");  
break;  
default :  
trace("other");  
break;  
}
```

Ma lenh tren qua don gian, khong can giao thich. ^^!

5. Loops

Vong` lap la cai don gian, chac ai cung biet roi`. Neu vong` lap ma ban con chua biet thi` nen hoc lai pascal chuc khong nen doc cuon sach nay`, tui luoi` ma :D.

```
for (var i:Number = 0; i < 3; i++) {  
trace("hello");  
}
```

Vong` lap nay thuc hien 1 cong viec tuong tu voi vong lap:

```
for (var i:Number = 3; i > 0; i--) {  
trace("hello");  
}  
while Loop
```

```
var num:Number = 0;  
while (num < .5) {  
num = Math.random();  
}
```

Vong lap nay co so lan khong biet truoc do ham Math.random() dung de lay 1 so ngau nhien $0 < x < 1$, DK num $< .5$ chua biet lan se dc tho man.

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE

Luu y voi vong while tranh truong hop lap vo han :

```
var flag:Boolean = true;  
while (flag) {  
    trace ("infinite loop");  
}
```

Cac vong lap ton kha nhieu tai` nguyen may tinh, do do' chi dung` vong lap khi can thiet va biet toi uu hoa vong lap.

6. Arrays

Cai nay cung qua ro rang` roi`, cac ban nen doc sach “24h hoc ActionScript” ve phan nay vi` minh` thay sach do' trinh` bay rat chi tiet.

7. Functions

```
function showMsg(){  
    trace("hello");  
}  
  
showMsg();
```

Ham tren thuc hien cong viec in ra man hinh output “hello”. Ham nay khong co tham so va khong tra ve gia tri nao ca. Sau day la 1 ham co tham so:

```
function showMsg(msg:String) {  
    trace(msg);  
}  
  
showMsg("goodbye");  
//goodbye
```

Tham so la **msg** khi thuc hien cau lenh **showMsg("goodbye")**; thi chuoi goodbye duoc truyen cho tham so **msg** va Ham **showMsg()** thuc hien cau lenh **trace(msg)**;

Con day la 1 vd ve ham tra ve` 1 gia tri:

```
function celToFar(cel:Number):Number {  
    return (9/5)*cel + 32;  
}  
  
trace(celToFar(20));  
//68  
  
function farToCel(far:Number):Number {  
    return (5/9)*(far - 32);  
}  
  
var celDeg:Number = farToCel(68);  
trace(celDeg);  
//20
```

Sau cau lenh **return** la gia tri ma ham se mang, no la cau lenh ket thuc cua 1 ham tra ve gia tri, tu day ham **celToFar(cel)** co the su dung nhieu 1 bien binh thuong AS, ta dung lenh **trace(celToFar(20))**; de in ra Giatri cua ham voi tham so duoc truyen gia tri 20 va KQ o cua so output la 68

```
function celToFar(cel:Number):Number {
```

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE J

return (9/5)*cel + 32;

}

Kieu Du lieu **Number** co the ghi lai la **void** neu ham khong tra ve gia tri nao.

8. Custom Objects

Phan nay minh cung hok ro~ lam (sach viet dai` wa'). Dai khai **Custom Objects** la mot bien du lieu co nhieu thuoc tinh khac nhau va no duoc dung thay cho mang o nhung kieu du lieu phuc tap co nhung bien co nhieu thuoc tinh khac nhau.(Nhu con ng` thi co cac chi? So nhu can nang, chieu cao, IQ...)

Vd:

```
var plane:Object = new Object();
plane.pitch = 0;
plane.roll = 5;
plane.yaw = 5;
ket hop voi ham:
function showPlaneStatus(obj:Object):void {
trace(obj.pitch);
trace(obj.roll);
trace(obj.yaw);
};
showPlaneStatus(plane);
```

Nho khai bao bang lenh **var Tenbien:Object = new Object()**; la dc, sau do gan' tiep cac thuoc tinh cua no.

Tu muc 4 à 8 minh luoi` viet vi` thay no de~ va dang lam giua chung thi` bi cup dien nen dang phai danh lai L, cang them nhac' nen minh chi viet ngan ngon thui, chac ai cung hieu duoc J .

9. this

Gia su ban co 1 movieclip co ten mc va de goi thuoc tinh chieu dai cua no' va ban goi no tu timeline chinh' cua Flash thi': vd 1:

this.mc.width (dang la o main timeline chinh' cua **mc**) Con neu nhu Ma lenh ban viet o trong timeline chinh' cua **MovieClip MC** nam trong flash thi phai viet lai , **vd 2: this.parent.mc.width**; nam trong ma~ lenh chinh' cua main timeline cua **mc.parent** duoc su dung khi chi mot doi tuong khac nam trong flash (co the la MovieClip khi dung parent thi ma~ lenh duoc nhay 1 LV)

Trong ca 2 truong hop tren thi **this** duoc xem nhu diem bat dau ma lenh (cua duong dan den Movie Clip can tham chieu), no' co the goi thuoc tinh chieu dai cua **mc** trong cung 1 LV (trong main timeline chinh' cua Flash-VD1) hay o lv cao hon (o? VD2-vd2). Flash hieu "parent" la "cha me" cua doan timeline chua' timeline chinh' dang chua' ma lenh(co LV thap hon) going nhu 1 gia dinh co nhieu thanh vien va the he, se co cha me ong ba` ... neu ban goi "cha me" (parent) thi` chi co "cha me" cua ban tra loi` (con` ong ba` thi` tat nhien la hok) . Voi' ban thay vi` goi "cha me cua toi" hoac "me cua cha me toi", ban cho the chi ra dieu do (co le thong qua dung` this) ro rang lam cho ban co huong dung dan.(chac la chi den dung MovieClip can chi)

(phan nay dich hok thoat nghia – xin trich nguyen van)

In both cases, **this** is a reference point from which you start your path. It is fairly common to drop the keyword when going down from the current scope (as in the first example), but it is required when going up to a higher scope (as in the second example). This is because Flash must understand what the parent is actually a parent of in order to start traversing through the hierarchy. Imagine a family reunion in which several extended family members, including

BIÊN SO N L I T SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE

cousins and multiple generations, are present, and you are looking for your mother, father, or grandparent. If you just said “parent,” any number of parents might answer. If you, instead, said “my parent” or “my mother’s parent,” that would be specific enough to get you headed in the right direction

Hien o cac vd tren, this dc xem la LV0 nhung neu dem **mc** su dung trong Moi truong flash thi Moi truong flash moi la LV0 con this trong timeline chinh cua **mc** la chi LV1, tuong tu trong timeline chinh cua 1 MovieClip chua’ trong **mc** thi do la LV2

10. Absolute versus Relative Addresses

Giong nhu HDH co cac thu muc (folder) hay cau truc file cua 1 website , ban co the tham chieu den 1 doi tuong trong flash bang duong dan truc tiep hoac cac duong dan co lien quan toi cac doi tuong co lien quan va den doi tuong can tham chieu(duong dan gian tiep). That de dang neu dung duong dan truc tiep den mot doi tuong da biet chinh xac bat ky trong flash, tuy nhien dieu do co ve cung nhac va se khong con dung neu ta thay doi bang duong dan gian tiep. Neu dung duong dan gian tiep thi doi tuong tro? Len ro nghia va co tinh mem deo khi viet chuong trinh.

Ta co the xem 2 vd sau:

Table 2-2. Absolute (from main timeline to nested movie clip)

ActionScript	Windows OS	Mac OS	Web Site
<code>root.mc1.mc2</code>	<code>c:\folder1\folder2</code>	<code>Macintosh/folder1/folder2</code>	<code>http://www.domain.com/dir/dir</code>

Table 2-3. Relative (from a third movie clip, up to the root, and down to the child of a sibling)

ActionScript	Windows OS	Mac OS	Web Site
<code>this.parent.mc1.mc2</code>	<code>..\folder1\folder2</code>	<code>../folder1/folder2</code>	<code>../dir/dir</code>

O ca 2 bang thi Ma lenh duoc viet o Main timeline cua flash va tham chieu den mc2 nam trong mc1 cua flash nhung cach viet khac nhau.

* Bang 2-2 thi bat dau tu timeline chinh’ cua **FLASH** (lv0) chi tiep den mc1 roi den mc2 (tang dan lv)
Ma lenh trace(this.mc1.mc2.x) à dua ra toa do x cua mc2 nam trong mc1.duoc dat o timeline chinh’ cua flash
* Bang 2-3 Theo minh` hieu thi` dat ma len o MovieClip thu 3 sau do up len chinh trinh` chinh’ (noi co mc1)va` down xuong MovieClip2 nam trong MovieClip 1

Phan nay minh` dich khong dung` vi` khong check duoc ma lenh nay` khi o nha` - Ai hieu duoc doan nay` xin lien he minh` thanh_vinh648@yahoo.com xin chan thanh` cam on

Much like a computer operating system’s directory, or the file structure of a web site, ActionScript refers to the address of its objects in a hierarchical fashion. You can reference an object address using an absolute or relative path. Absolute paths can be easy because you most likely know the exact path to any object starting from the main timeline. However, they are quite rigid and will break if you change the nested relationship of any of the referenced objects. Relative paths can be a bit harder to call to mind at any given moment, but they are quite flexible. Working from a movie clip and going up one level to its parent and down one level to a sibling will work from anywhere—be that in the root timeline, another movie clip, or nested even

B N D CH NÀY C VI T B I HUYETSATHIENHA – thanh_vinh648@yahoo.com
12/1 – THPT QU C H C HU - B N D CH L U HÀNH TRÊN <http://WWW.VNFX.COM>

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER 'S GUIDE J deeper—because the various stages aren't named.

Table 2-2 and Table 2-3 draw parallels to the operating system and web site analogies:

Graphics and Interaction

Chapter 3: Properties, Methods, and Events

1. Properties

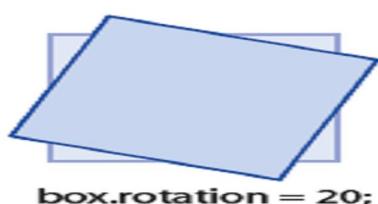
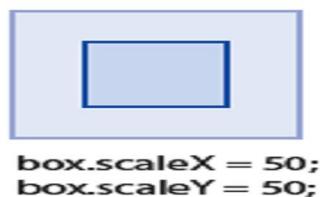
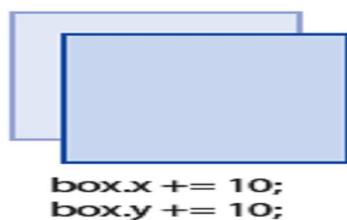


Table 3-1. Movie clip properties

Description	Property	Syntax for Setting Value	Units and/or Range
Location	x, y	box.x = 100; box.y = 100;	pixels
Scale (1)	scaleX, scaleY	box.scaleX = .5; box.scaleY = .5;	percent / 0-1
Scale (2)	width, height	box.width = 72; box.height = 72;	pixels
Rotation	rotation	box.rotation = 45;	degrees / 0-360
Transparency	alpha	box.alpha = .5;	percent / 0-1
Visibility	visible	box.visible = false;	Boolean

La nhung thuoc tinh cua Doi tuong.
VD: Nhung thuoc tinh cua MovieClip

VD cau lenh don gian de thay doi thuoc tinh cua MC:

box.rotation += 20;

trong **box.alpha = .5** (tuc lagia tri box.alpha = 50 trong AS2).

2. Events

La nhung su kien trong flash, moi su kien gan voi 1 tho` diem va trong tho` diem do doi tuong thuc hien nhung cau lenh nhat dinh. Cai nay cung going nhu trong AS2, mot so VD cho viec goi su kien trong main timeline cua flash: **myButton.onRelease = function(){lenh};** la su kien nhap chon nut lenh myButton hay ma lenh cung thuc hien cau lenh tren song no dc viet trong Main timeline cua myButton: **on(Release)=function(){lenh};** Nhung event do deu la event truc tiep khi xuat hien 1 su kien. Duoi day la cach dung` event gom nhieu su kien lien tiep nhat moi thuc hien cau lenh.

a. Using Event Listeners

Co so cua **Event Listeners** rat don gian, hay tuong tuong ban dang noi thuyet trinh voi 100 nguoi ma chi co 1 nguoi la thinh' gia

T B I HUYETSATHIENHA – thanh_vinh648@yahoo.com

THU - B ND CH L U HÀNH TRÊN <http://WWW.VNFX.COM>

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE J

tra loi lai 1 cau hoi ro rang` cua nguoi thuyet trinh`. Trong truong hop do chi co 1 ng da noi va nghe theo su kien do' va hanh dong theo nhung loi` huong dan da duoc cung cap khi su kien xay ra.

(doan nay minh hok hieu lam, xin dc trinh lai nguyen van:

The concept of event listeners is pretty simple. Imagine that you are in a lecture hall that holds 100 people. Only one person in the audience has been given instructions about how to respond when the lecturer asks a specific question. In this case, one person has been told to listen for a specific event, and to act on the instructions provided when this event occurs.

)

Cac cau lenh cua **Event Listeners** cung vay, no se thuc hien khi 1 hay nhieu su kien xay ra: Khi co mot su kien xay ra, ban muon thuc hien mot hay nhieu ham ban nen su dung **Event Listeners**. ham nay co doi' so se nhan thong tin tu su kien ma ban da goi no. cho phep ham su dung mot phan du lieu cua su kien dang dien ra va thuc thi ma lenh cua no Tom lai addEventListener() la mot Phuong phap (Methods) da duoc dung` de nhan biet 1 su kien va chi dinh 1 ham de giao quyet cho su kien do, va cho rang hanh dong do dang duoc thuc hien. Tro lai voi vd button example, hay tao 1 nut co ten *rotate_right_btn* va them mot event listener cho methods mouse up. Tao them mot MovieClip co ten **box**

```
rotate_right_btn.addEventListener(MouseEvent.MOUSE_UP, onRotateRight);
function onRotateRight(evt:MouseEvent):void {
    box.rotation += 20;
}
```

O dong lenh dau tien ban da bat dau` voi ten cua nut lenh va them chuc nang (methods) **addEventListener()** chuc nang nay yeu cau bat buoc phai co 2 tham so: 1 la su kien ban se them vao, va cai kia la su kien duoc them vao.

Tu dong 2 à 4 la viet ham cho chuc nang **onRotateRight** se lam` quayMovieClip **Box** mot lan click vao` nut *rotate_right_btn* thi **Box** se quay 20°

Bay gio` hay thay MOUSE_UP bang` MOUSE_DOWN va xem xet su khac biet. Thu vai` lan` ban se thay su khac biet giua chung. Voi` MOUSE_UP thi khi ban click bao button vat ha chuot thi` cau lenh moi' co' hieu luc, con voi MOUSE_DOWN thi chi can ban click bao button thi ma lenh da duoc thuc hien.

Ngoai ra ban co the viet them chuc nang ENTER_FRAME cho Su kien Event (cau lenh se duoc thi hanh` moi~ khi flash duyet qua 1 frame) hay chuc nang KEY_UP cua su kien

Tham so thu 2 trong **addEventListener()** la ten ham duoc them vao khi su kien duoc xay ra. Sau do o dong lenh 2 la dinh nghi cua ham **onRotateRight** thuc hien chuc nang lam xoay MovieClip **Box** 20° moi khi click vao button. Ham nay khong tra lai gia tri len co kieu :**void** .

Co nhung thu ma doi' so cua ham khong the giao thich day du duoc khi no nhan mot su kien. Khong giong nhu nhung ham binh` thuong` khac ham listener (listener function) doi so trong ham ham listener la bat buoc phai co . trong VD tren **evt** la doi' so cua ham **onRotateRight** la nham vu cua no la nhan thong tin tu cac yeu to gay ra su kien (la con chuot). Doi so phai cung kieu du kieu voi su kien dang duoc cho` doi. La se xay ra. Trong mot so truong hop vi` chung ta dang cho` doi. Su kien **MouseEvent** nen day cung la kieu du lieu cho doi so (la MouseEvent).

Mot so VD khac, chung ta se xem xet truong hop co 2 ham khi goi 2 chuc nang **addEventListener** : Truoc het hay tao 2 MovieClip co ten myMovieClip va myMovieClip2, go~ doan ma nay vao timeline chinh'

BIÊN SO N L I T SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE

cua flash.

```
myMovieClip.addEventListener(MouseEvent.MOUSE_DOWN, onStartDrag);
myMovieClip.addEventListener(MouseEvent.MOUSE_UP, onStopDrag);
function onStartDrag(evt:MouseEvent):void {
    evt.target.startDrag();
}
function onStopDrag(evt:MouseEvent):void {
    evt.target.stopDrag();
}
```

Trong vd nay, 2 event listeners da duoc dang ky su dung cho MovieClip **myMovieClip**. Mot su kien cho **mouse down** va mot su kien cho **mouse up**. Chung trong khong khac nhau nhieu, tuy nhien thuoc tinh **target** cua hai ham la thu duoc tim` thay o doi so cua ham, no la cai dung` de nhan ra yeu to (thong tin) duoc nhan ra trong se kien MouseEvent. Dieu nay cho phep ham` o dong` lenh thu 3 co the lam dragging MovieClip **myMovieClip** dragging (keo le) khi dang click va giu con chuot, vat duoc tha ra (het dragging) khi tat ha con chuot. Cach tiep can co dac diem chung la rat huu dung va no lam cho viec su dung ham tro len linh hoat. Ham co the hoat dong mot cach chinh xac khi chuot duoc click va duoc tha (pass), noi cach khac ham co the bat dau` hay dung` viec keo tha cua MovieClip bat ky` den noi nao do bang ham` vua` them vao`.

Sauk hi ban da viet ma lenh tren, ban co the ap dung no bao mot MC khac bat ky`

Vd ap dung vao myMovieClip2:

```
myMovieClip2.addEventListener(MouseEvent.MOUSE_DOWN, onStartDrag);
myMovieClip2.addEventListener(MouseEvent.MOUSE_UP, onStopDrag);
```

Using Mouse Events to Control Properties

(su dung nhung su kien cua chuot de lam mot bang dieu khien thuoc tinh cua doi tuong).

Dai khai' sau khi lam xong thi` ta se co cai' tuong tu nhu cai nay`:

1

PROPERTIES and MOUSE EVENTS

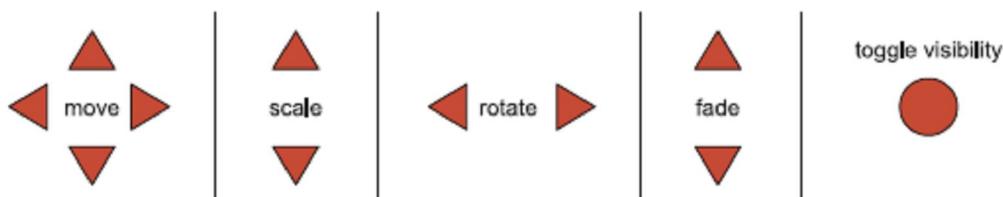


Figure 3-2. Layout of the props_events.fla file

Tao mot MovieClip co ten la Box

Lan luot tao cac nut lenh sau: move_left_btn, , move_right_btn, move_up_btn, move_down_btn.

Them doan ma nay vao

```
move_left_btn.addEventListener(MouseEvent.MOUSE_UP, onMoveLeft);
move_right_btn.addEventListener(MouseEvent.MOUSE_UP, onMoveRight);
move_up_btn.addEventListener(MouseEvent.MOUSE_UP, onMoveUp);
move_down_btn.addEventListener(MouseEvent.MOUSE_UP, onMoveDown);
```

sau do viet cac ham` them vao`:

```
function onMoveLeft(evt:MouseEvent):void {
box.x -= 20;
};
function onMoveRight(evt:MouseEvent):void {
box.x += 20;
};
function onMoveUp(evt:MouseEvent):void {
box.y -= 20;
};
function onMoveDown(evt:MouseEvent):void {
box.y += 20;
};
```

Tuong tu tao cac nut lenh scale_up_btn, scale_down_btn, rotate_left_btn, rotate_right_btn, fade_in_btn, fade_out_btn, toggle_visible_btn.

Them chuc nang addEventListener vao cac doan sau:

```
scale_up_btn.addEventListener(MouseEvent.MOUSE_UP, onScaleUp);
```

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER 'S GUIDE

```

scale_down_btn.addEventListener(MouseEvent.MOUSE_UP, onScaleDown);
rotate_left_btn.addEventListener(MouseEvent.MOUSE_UP, onRotateLeft);
rotate_right_btn.addEventListener(MouseEvent.MOUSE_UP,
onRotateRight);
fade_in_btn.addEventListener(MouseEvent.MOUSE_UP, onFadeIn);
fade_out_btn.addEventListener(MouseEvent.MOUSE_UP, onFadeOut);
toggle_visible_btn.addEventListener(MouseEvent.MOUSE_UP,
onToggleVisible);

```

va them dinh nghia cac ham` ma ta se su dung

```

function onScaleUp(evt:MouseEvent):void {
box.scaleX += 0.2;
box.scaleY += 0.2;
};
function onScaleDown(evt:MouseEvent):void {
box.scaleX -= 0.2;
box.scaleY -= 0.2;
};
function onRotateLeft(evt:MouseEvent):void {
box.rotation -= 20;
};
function onRotateRight(evt:MouseEvent):void {
box.rotation += 20;
};
function onFadeIn(evt:MouseEvent):void {
box.alpha += 0.2;
};
function onFadeOut(evt:MouseEvent):void {
box.alpha -= 0.2;
};
function onToggleVisible(evt:MouseEvent):void {
box.visible = !box.visible;
};

```

XONG!.

3. Methods (Phuong phap').

Methods la mot dong tu` trong ngon ngu lap trinh AS, dung de cho biet tung doi' tuong trong flash se co nhung cong viec gi phai lam. VD nhu ban co the bat 1 MovieClip dung` viec choi hoat hinh bang cach dung **stop()** method. Gieng nhu thuoc tinh(property) Methods luon xuat hien dung luc va thich hop voi cu phap cua cac cau lenh trong AS. Theo sau 1 doi tuong ta co the goi 1 methods, lay VD nhu tren , ta co the lam MovieClip **Box** stop bang` cau lenh

Box.Stop();

Using Keyboard Events to Call Methods

(su dung su kien cua ban` phim' de goi chuc nang Methods)

Viec su dung su kien ban` phim' cung going nhu su dung su kien cua chuot, chi khac mot dieu : Muc

BIÊN SO N L IT SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE J

tieu cua su kien listener se khong thuong xuyen la doi’ tuong ma ban muon muon van dung (**The target of the event listener is not frequently the object you wish to manipulate.**) Khi ta lam viec voi text ... Khi ban dieu khien mot MovieClip chinh’ ban than no (stage) lai thuong su dung de kiem soat de tiep thu cua su kien tu`ban` phim’

(**When working with text, the text field being manipulated may, indeed, serve well as the target of the keyboard events. When controlling movie clips, however, the stage itself is often a useful, centralized recipient of the keyboard events.**)

Them mot su kien listener vao stage (chuong trinh chinh) co nghia la ban co them xu ly them voi’ m i su kien voi mot listener don gian.. **Listener don gi n** có m t c u trúc n gi n. Chung ta ā làm quen v i các câu lệnh switch va **if/else-if** chapter 2. ó c ng là một ki u c u trúc i u ki n n gi n.Chúng ta s b t u them vào 1 listener vào trong Stage, trong tr ng h p này, chúng ta s chú ý n s ki n **key down**, su kien nay se tiep nhan moi phim’ duoc danh tu ban phim’ bang cach them vao cho nó ch c n ng **onKeyPressed** trong stage.

u tiên hãy t o 1 MovieClip có tên là box, trên timeline nh n F6 t o nhi u key frame k ti p nhau m i key frame b n box m t tr ng thái khác nhau(nh v màu s c, kick th c,... d quan sát)

stage.addEventListener(KeyboardEvent.KEY_DOWN, onKeyPressed);

nh ngh a hàm **onKeyPressed** :

```
function onKeyPressed(evt:KeyboardEvent):void {  
    switch (evt.keyCode) {  
        case Keyboard.ENTER:  
            box.play();  
            // làm cho Box play  
            break;  
        case Keyboard.BACKSPACE:  
            box.stop();  
            // làm cho box d ng l i  
            break;  
        case Keyboard.LEFT:  
            box.prevFrame();  
            //làm box ch y l i tr ng thái (key frame) tr c ó  
            break;  
        case Keyboard.RIGHT:  
            box.nextFrame();  
            //làm box ch y t i tr ng thái (key frame) sau ó  
            break;  
        case Keyboard.SPACE:  
            box.gotoAndStop(3);  
            //làm box ch y l i tr ng thái (key frame) th 3  
            break;  
        default:  
            trace(evt.keyCode);  
            // còn l i thì hi n ra mã ASCII c a ký t c a s output  
    }  
};
```

Chú Y:M t s phím s không a ra mã ASCII c a nó b i vì ó là phím t t c a flash, khi nhân phím ó thì flash t ng nh m la b n ang dùng phím t t ch không ngh là b n ang b m phím ó ch y th ch ng trình, do ó phím có hi u l c thi b n

B ND CH NÀY C VI TB I HUYETSATHIENHA – thanh_vinh648@yahoo.com

12/1 – THPT QU CH CHU - B ND CH L U HÀNH TRÊN <http://WWW.VNFX.COM>

vào Control>Disable Keyboard Shortcuts sau đó Control>Test Movie command..

Event Propagation

Sự kiện sẽ được truyền đi từ 'Chương trình chính' (stage) đến _root (gốc LV0 của chương trình) đến một folder mc1 nào đó chứa movieclip mc2 và mc3, khi sử dụng event propagation thì các MovieClip từ stage đến mc1 hoặc mc2 đều có thể sử dụng chung các hàm khi định nghĩa chung xảy ra thuộc một hay nhiều event nào đó.

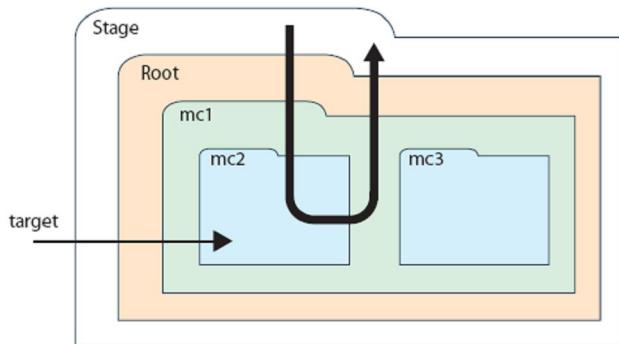


Figure 3-3. Event propagation process

(Sự truyền Sóng nhanh)

Đó! trong trang display list là một phần của sự kiện xảy ra tiếp theo (tương tự), thông thường được chuyển đến theo **toas event propagation**. Khi kết quả nham nhở của một events đặc biệt xảy ra, bao gồm mouse và key events, thì trong display list, event sẽ không gửi đi được một cách trực tiếp **hay nói cách khác nhung~ event sau khi ta dung` event propagation thi` ta co the ap dung no cho nhieu event cua nhung~ movieClip khac` ma` khong mat cong dinh nghia lai ham` thuc hien trong event do`**.

nhưng event target. Tất nhiên chúng sẽ gửi tới display list, và event propagates từ phía trên của danh sách (list down) đến với event target, **and then bubbles(works its way) back up through the display list again**. Consider two movie clips (mc2 and mc3) within a movie clip (mc1) that is on

the stage. Next, imagine that the target of the event is the nested movie clip, mc2. When the desired event occurs, it is not dispatched directly to mc2, but

rather to the display list. First, the stage receives the event, then any relevant loaded SWFs (including the root timeline, in this example), then the parent movie clip, mc1, and then the target of the event, mc2. After the event is received by the target, it then propagates back up through the display list to mc2, root, and stage. Figure 3-3 depicts the process, showing a mouse event dispatched to the top of the display list, the stage, making its way through the root timeline and parent movie clip until it reaches the event target, and then bubbling back up through the display list again.

Sau đây là một ví dụ về event propagation:

```
folder0.addEventListener(MouseEvent.MOUSE_OVER, onFolderOver);
folder0.addEventListener(MouseEvent.MOUSE_OUT, onFolderOut);
folder1.addEventListener(MouseEvent.MOUSE_OVER, onFolderOver);
folder1.addEventListener(MouseEvent.MOUSE_OUT, onFolderOut);
function onFolderOver(evt:MouseEvent):void {
    evt.target.alpha = 0.5;
}
function onFolderOut(evt:MouseEvent):void {
    evt.target.alpha = 1;
}
```

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER 'S GUIDE ↗

Hai movieClip folder0 va` folder1 se bi lam mo` di 50% khi re chuot len tren MovieClip do'.

Ham` onFolderOver, onFolderOut nay` co the su dung rong rai cho nhieu MovieClip khac nhau.

Tuy nhien neu 1 MovieClip giong nhau nhu co nhieu phien ban khac nhau (instance name khac nhau)cung muon su dung chung 1 hay nhieu ham` khi dang dien ra 1 su kien trong khi lai co nhieu MovieClip khac lai khong muon su dung nhung ham` do' khi 1 su kien do' xay ra.Lay VD muon tao ra 1 so MovieClip co chuc nang trong 2ham`**onFolderOver, onFolderOut**, nhung khong phai toan bo MovieClip trong file flash cua ban cung muon 2 ham` nay`, vay ban co the dat nhung MovieClip muon` chuc nang trong event propagation bang cach dat doan ma lenh sau vao frame 1 tren timeline chinh` trong MovieClip can` su dung ham` (**chu' khong phai timeline chinh` cua flash dau nha**) nho' chuyen sharp trong MovieClip thanh` 1 MovieClip nua va` dat ten cho no la` **folder_group**.

```
folder_group.addEventListener(MouseEvent.MOUSE_OVER, onFolderOver);
```

```
folder_group.addEventListener(MouseEvent.MOUSE_OUT, onFolderOut);
```

```
function onFolderOver(evt:MouseEvent):void {
```

```
    evt.target.alpha = 0.5;
```

```
}
```

```
function onFolderOut(evt:MouseEvent):void {
```

```
    evt.target.alpha = 1;
```

```
}
```

Ung` dung: ban co the tao ra 1 flash co hieu ung lam` mo` 1 tap hop MovieClip khi chuot re len trong Man` hinh` hien thi



Frame and Timer Events

(Frame va` su kien cho thoi` gian).

Frame event

Frame event giong voi su kien onEnterFrame cua AS2, chi khac ve` cu' phap' cau lenh thoai. Xin trinh dan VD:

```
stage.addEventListener(Event.ENTER_FRAME,onFrameLoop);
```

```
function onFrameLoop(evt:Event):void {  
    cycle.x = mouseX;  
    cycle.wheel.rotation = mouseX;  
}
```

Ham tren se lam cho MovieClip cycle di chuyen theo toa do x con chuot va` xoay mot movieClip co ten la wheel nam trong MovieClip cycle (co thuoc tinh goc' rotation bien thien theo vi tri' toa do x con chuot)

Ban co the viet lai ma lenh nay trong AS2:

Cai' nay` chi? Mang tinh' chat tuong trung thoai ^^!

```
Cycle.onEnterFrame = function(){  
    this.x = _root._xmouse;  
    this.wheel.rotation = _root._xmouse;  
}
```

Timer Events

Time Event la` mot su thay the cua Enter frame de Ma lenh co the hoat dong mot cach tuan` hoan` khi su dung su kien ma tho` gian duoc quy dinh san, Mac du` lua chon Enter Frame la mot trong nhung su lua chon de dang` song, chung ta van co the kiem soat no bang cach dieu` chinh chi so' **frames per second**.

Trong nhung phien ban truoc cua AS thi` su dung chuc nang **setInterval()** de thiet lap tho` gian xay ra 1 ham` (hoac cau lenh) va` chuc nang **setTimeout()** de dinh tho` gian ket thuc lam` viec cua 1 ham hoac cau lenh nao` do'.

Nhung trong AS3 tat ca ban chi can` thuc hien cac Methods (chuc nang) cua lop' (**class**) timer san~ co' trong flash

Cu phap don gian cua mot bien time:

```
var timer:Timer = new Timer(delay:Number, repeatCount:int);
```

trong do: delay la tho` gian ma` ham` se dung` lai de thuc hien.(gia tri bat buoc)

repeatCount: la` so lan lap lai tho` gian do' à so lan lap cau lenh trong ham` dang duoc thuc thi (**gia tri neu de trong' thi` AS se hieu la` lap khoang tho` gian tren vo han cho den' khi chuong trinh` bi dong' boi nguoi` dung`.**)

Su kien tho` gian ma` chung` ta nghien cuu se la cach lam mot khoang tho` gian nao` do', cach ket thuc tho` gian van hung viec co the lam trong khoang tho` gian do' kho khoang tho` gian cua chung ta van chua ket thuc'.

Lay' VD sau:

```
var timer:Timer = new Timer(1000);  
//so 1000 o tren duoc tinh theo milliseconds (mili giay)  
// khai bao bien time thuoc kieu Timer va co tho` gian la 1s.  
timer.addEventListener(TimerEvent.TIMER, onTimer);  
//them ham` onTimer vao` su kien TimerEvent.TIMER.  
timer.start();  
// Dong` lenh quan trong nhat la dong` 3 vi bat dau` tinh tho` gian (va ket thuc sau do' 1 //giay)  
flash khong tu dong lam dieu nay`.  
function onTimer(evt:TimerEvent):void {  
    watch.rotation +=5;  
    //cai' MovieClip watch se quay 5° mot giay (neu ban de y' thi` day la kim giay cua dong` ho)
```

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE J

}

Ngoai ra ban su dung methods(chuc nang) stop() va reset() de lam dung` thoi` gian hoac khai dong lai thoi` gian da dat.

```
timer.stop();
timer.reset();
```

Removing Event Listeners

(xoa` bo mot Event Listener)

De xoa bo 1 Event Listener() chung ta dung` chuc nang **removeEventListener()** ma` tham so' (la nhung thu' nam trong dau' ngoac don ()) se~ la nhung tham so cua **Event Listener()** ma` ban muon' loai. Bo?. VD o tren ta co the loai. Bo MovieClip watch sau khi no' quay duoc 25° bang ma~ lenh sau:

```
var timer:Timer = new Timer(1000);
timer.addEventListener(TimerEvent.TIMER, onTimer);
timer.start();
function onTimer(evt:TimerEvent):void {
watch.hand.rotation +=5;
if (watch.hand.rotation >= 25) {
timer.removeEventListener(TimerEvent.TIMER, onTimer);
}
}
```

Hoac don gian hon neu ban de y rang` Ham onTimer hoat dong trong 1 giay va` trong 1 giay do' thi` MovieClip watch se quay duoc 5° vay de MC Watch quay duoc 25° roi` dung` lai thi` co the de bien time khong lap vo han ma` lap 5 lan` bang` ma lenh sau:

```
var timer:Timer = new Timer(1000, 5);
```

Tren day la nhung cach de loai bo Event Listener , chung ta se thao luan thiеп nhanh van de` nay` bang` cach them vao` cac kick ban khac su dung cho cong viec loai bo chung' trong bang “**Garbage Collection**” day la nhung cach luyen tap tot de loai bo Event Listener , nhung su kien ban khong muon chung' ton` tai lau.

“**Garbage Collection**” la bang nhung tham so can nho` cho viec su dung event Listener.

Co' 3 tham so ban co the them vao Event Listener, nhu VD: sau:

```
eventTarget.addEventListener(EventType.EVENT_NAME, eventResponse,
useCapture:Boolean, priority:int, weakReference:Boolean);
```

Day la cu' phap chung thoi nha, tui se giao thich nghia ngay o duoi'.

Phan` gach chan la nhung tham so moi` duoc them vao`

Tham so dau tien useCapture cho phép ban dieu khien duoc Event Listener truoc khi no dat duoc muc dich cua no' (if set to **true**) or 1 lan khi no da dat duoc muc dich cua no (if set to **false** gia tri nay mac dinh la **false**) or se co 1 thong bao' loi~ dua ra ngoai man` hinh` . Ban se su dung gia tri false cho tham so nay` trong phan nhieu` truong` hop (phan lon thoi` gian). Cai nay` thay cung kho` hieu – xin trich lai nguyen van:

The first optional parameter, useCapture, allows you to handle the listener event before it reaches its target (if set to true) or once the event has reached its target (if set to false) or is bubbling back up through the display list. The default (false) is to react to all events captured at or after the event reaches the target, and this is the configuration you will likely use most of

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE

the time.

Tham so thu 2 duoc chon , **priority** dung` de lua chon quyen uu tien se xay ra listener nao` (ham` nao` se duoc goi) khi co nhieu` listener (tuc la cac ham` da duoc dinh nghia) cung` xay ra trong 1 su kien duoc goi.

VD: Khi ban thuc' day(event), ban phai lua chon giua danh' rang (listener) va` rua? mat (listener) xem cong viec nao` duoc uu tien truoc. – cai' nay` tui VD chu' sach hok co dau J - gia tri mac dinh (default) cua no' la 0 se giup chung' ta thuc hien cau lenh duoc trong nhieu hoan` canh khac nhau.

Nguyen van:

The second optional parameter, priority, allows you to order the execution of multiple listeners set to respond to the same event in the same phase. This, too, is unlikely to be an issue, and the default parameter of 0 will serve you well in the vast majority of circumstances

Con` tham so thu' 3 weakReference – chung ta nen di sau vao` tham so nay` de ban co the hieu va` su dung.Viec su dung tham so thu 3 nay thuc ra rat de dang`. Gia tri mac dinh la **false** va` ban co the thay doi no' thanh **true** de tien su dung. Khi thay doi thi` dau tien tham so thu' 1 va 2 can phai duoc dua vao` trong cu' phap truoc da de flash biet rang ban duoc muon thu thiet dat tham so nao`.(di nhien tham so 1 va` tham so 2 ban nen de la **false** va` O cho moi thiet dat truoc khi thiet lap tham so thu 3 la **true**) co nghia chung ta se thiet lap theo cau' truc' nhu sau:

**eventTarget.addEventListener(EventType.EVENT_NAME,
eventResponse, false, 0, true);**

Neu ban co thoi quen su dung cu' phap moi khi viet chuong trinh` (nghia la viet het tham so cua addEventListener() chang han J) thi` ban se viet chuong trinh` cham lai va` kho co the kiem soat tot bo nho' cua may' tinh'. Nho' rang` day khong phai la mot substitute (su thay the') cho viec loai bo mot listener duoc dien ra ro rang` . Tuy nhien ban co the lam` cai' nay` de du phong` hoac de tap luyen viec viet code cua ban thi` cung~ co the chap nhan duoc. Ban co the tham khao tai blog <http://Www.gskinner.com / blog> hoac co the xem them luong` su kien (flow event) o chapter 12 va 21 o cuon sach *Essential ActionScript 3.0*.

Phan` nay minh` dich cung thay kho' hieu, xin trich lai nguyen van: (cac ban thong cam – minh dich kem' ma`)

The third optional parameter, **weakReference**, is the option we want you to understand and start using. In a nutshell, this helps with memory management, in the event that you're not careful about removing unneeded listeners.

Briefly, in ActionScript 3.0, memory management that you do not explicitly control is handled behind the scenes by the garbage collector. When you are no longer referencing an object in your application, it is marked for cleanup, and the garbage collector periodically sweeps through your application discarding unneeded items, freeing up memory along the way. However, if a reference to an object remains, the garbage collector can't know that the object should be purged from memory.

Try as we might to be good, it's not uncommon for developers to forget to remove event listeners in their code (see the section "Removing Event Listeners" in this chapter). However, a distant next-best thing is a weakly referenced listener. Simply put, weakly referenced listeners aren't supervised by the garbage collector and, therefore, don't have to be manually marked for removal. If only weak references to an object remain after you have finished using it, then the object is eligible for collection. Using this option is very simple. All you need to do is change the **weakReference** setting of the **addEventListener()** method from its default value of **false**, to **true**. Because it's the third optional parameter, values for the first and second

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER 'S GUIDE J

parameters must be included so Flash knows which parameter you are trying to set. You will rarely need to change those values, so you can use their aforementioned defaults (**false** for **useCapture** and **0** for **priority**).

So, our preference, and the convention we will use hereafter in this book, is to use the **addEventListener()** method with this syntax:

```
eventTarget.addEventListener(EventType.EVENT_NAME,  
eventResponse, false, 0, true);
```

If you get in the habit of using this syntax, you will be far less likely to run into memory management problems due to lax code maintenance. Remember, this is not a substitute for removing your unneeded listeners explicitly. However, it's a backup plan, and a best practice that is easy to adopt.

Additional discussion of the event flow—including event phases, setting listener priority, stopping propagation along the way, manually dispatching events, and more—is featured on the companion web site. Flash developer Grant Skinner also wrote a helpful series of articles on resource management on his blog (<http://www.gskinner.com/blog>) that got us thinking about this in the first place. Finally, event flow is discussed in depth in Chapters 12 and 21 of our resource book of choice, *Essential ActionScript 3.0*.

Chapter 4 The Display List

(danh sach cua man hinh` hien thi)

The Sum of Its Parts

(tong cua nhung thanh phan` cua no' (Display List))

Mot so thanh phan` cua man` hinh` hien thi

- _ Shape: hinh` dang cua 1 thu gi` do' (hinh` chu nhat, duong thang...)
- _ Text: chu viet duoc bang cong cu Type (T)
- _ MovieClip: Co the la mot bitmap duoc convert qua MC
- _ Bitmap: hinh` anh duoc dua vao` flash

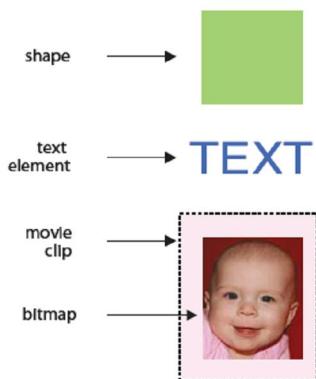


Figure 4-1. The visual layout of the sample file

Và chung

duoc chia theo trinh` tu nhu sau:

Hình 4.2 cho th y r ng màn hình li t kê
c a h s ó.

Tai dinh? Cua so do` tren la stage (chuong
trinh` chinh flash – swf) va` ban co the
truy cap den stage bang` nhieu cach khac
nhau tu nhieu` doi` tuong khac nhau tren
man` hinh` thi` stage van la nen` tang? Va`
chung` ta xay dung moi thu` tren man`
hinh` (display) tu` no` va` no` ho tro

B ND CH NÀY C VI TB II
12/1 - THPT QU CH CHU - B

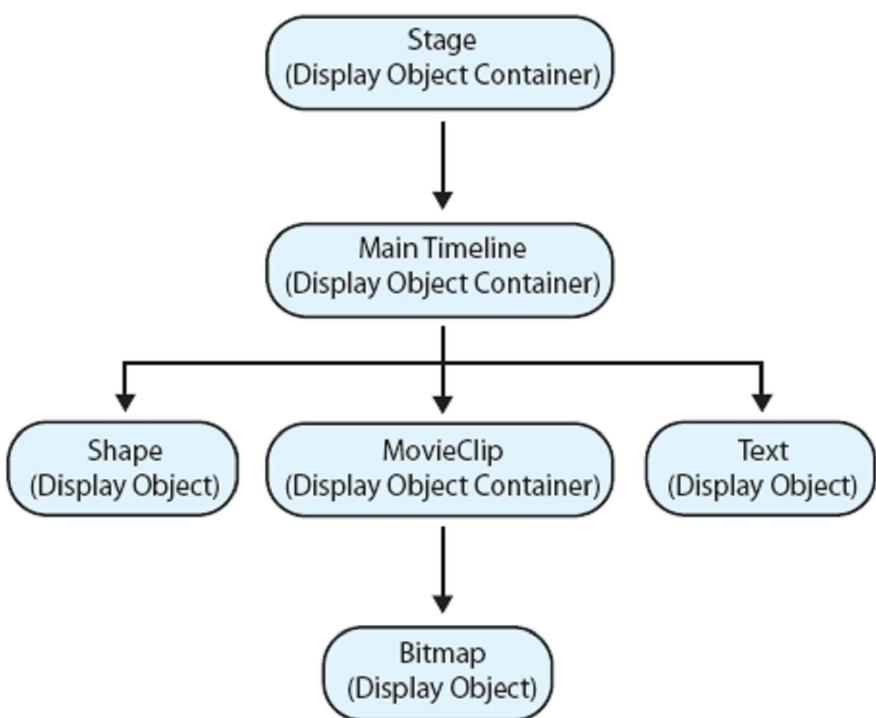


Figure 4-2. The display list of the sample file

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE J

container (doi’ tuong chua’ dung. Trong no’ – stage).

VD: hien nhien stage chua trong no’ nhieu` movieClip cua ch ng trình mà chúng ta t o ra ho c nhanh bitmap mà chúng ta a vào.

Tuong tu Main Timeline va` MovieClip cung co’ nhung doi’ tuong khac nhau ben trong no’

VD: MovieClip có th có các movieClip nam` ben trong no’ ma` chúng ta a h c cách truy nh p n chúng chapter 2.

Ti p theo là chính timeline, Mà c ng là ê` c p ns d ng bi n th hi n i t ng màn hình g c(**root** display

object instance variable). (Nhìn th y sidebar, “_root versus root” cho nhanh information.) m t h s flash yêu c u m t timeline bên trong ó có **all other assets are contained**. B i vì s truy n lan s ki n (event propagation), th t chung s d ng timeline chính t i m t v trí thêm nh ng event listener m i ang vi t nh ng o n mā trong timeline. Trong ng c nh ó, timeline chính tiêu bi u là referenced s d ng **this identifier**” (bi t chi ti t h n v nh ng s truy c p n listener trong s ki n listener ho c s kiên propagation, hãy nhìn xem Ch ng 3. bi t chi ti t h n v **this**, hãy xem Ch ng 2.)

Next is the main timeline, which is also referenced using the **root** display object instance variable. (See the sidebar, “_root versus root” for more information.) Like the stage, a Flash file requires one timeline within which all other assets are contained. Because of event propagation, it is common to use the main timeline as a location to add event listeners when writing scripts in the timeline. In that context, the main timeline is typically referenced using the **this** identifier, as in “this object being currently referenced within the context of the script.” (For more information about event listeners and event propagation, see Chapter 3. For more information about **this**, see Chapter 2.)

_root versus root

B n có th a hi u nh ng c n bi n ph i tránh s d ng nhung bien cho toàn ch ng trình Bi n _root trong nh ng phiên b n tr c C a ActionScript. ó b i vì giá tr c a bi n tùy thu c vào s thay i c a ch ng trình. Tr c ActionScript 3.0, bi n _root tham chi u t i timeline chính c a file SWF . không quan tâm file SWFs c t i nh th nào và bao nhiêu.

_root t ng ng c a m t a ch tuy t i, nh vi c vi n d n m t hình nh trong m t web-site VD: http://Www.yourdomain.com / hình nh, ho c m t ng d n n th m c v máy tính c a b n nh C:\directory\file

Vì _root là m t a ch tuy t i, Trong ActionScript 3.0, màn hình li t kē thay i mà th nh hành lôgic.

root bây gi là m t bi n th hi n c a i t ng màn hình(display object), và không ph i lúc nào c ng tham chi u t i timeline chính.

Nó s tr lên thích h p cho nhanh tr ng h p.root (c a) m t movie load trong m t SWF, Là c ng nh th n u Movie c load vào trong m t file SWF A. C ng t ng t th cho root load vào trong SWF B, Li u có th nōs ng m t mình ho c c load vào trong SWF C, Vân vân.

_root versus root

You may have heard you should avoid using the global

_root variable in prior versions of ActionScript. That’s because the value of the variable was subject to change. Before ActionScript 3.0,

the _root variable referred to the timeline of the original host SWF no matter how many SWFs got loaded.

BIÊN SO N L IT SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE J

_root was the equivalent of an *absolute* address, like referring to an image in a web site as *http://www.yourdomain.com/image*, or a file on your computer as *C:\directory\file*, instead of a more flexible *relative* address such as “image” (or “..*/image*,” for example, if you needed to traverse directories first). Because _root was an absolute address, if the file in which you

used the global variable was loaded into another file, the variable was redefined to become the timeline doing the loading, rather than your original file. This was often not initially intended and would break many object path references that originated with _root. In ActionScript 3.0, the display list changed that prevailing logic. root is now an instance variable of the display object, and doesn't

always refer to the main timeline. It's relevant to the context in which it's used so it behaves more like a relative address and no longer changes just because your SWF is loaded into another SWF. The root of a movie clip in SWF A, is the same if it stands alone or is loaded into SWF B. The same goes for the root in SWF B, whether it stands alone or is loaded into SWF C, and so on.

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER 'S GUIDE

Display List Classes

(l p danh sách Màn hình hi n th)

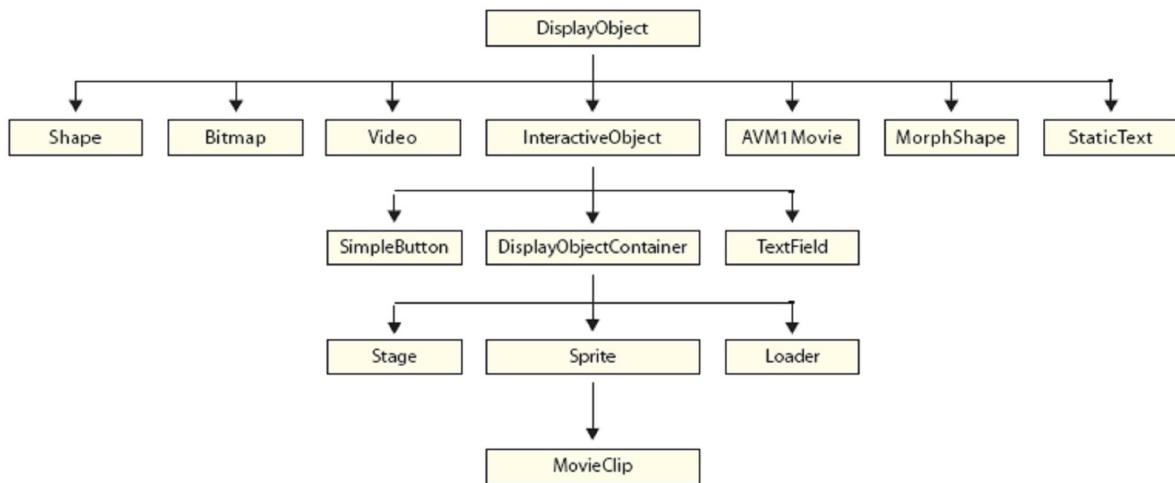


Figure 4-3. The DisplayList classes

Chúng ta ã th o lu n v cách s d ng l p (class) ch ng l và chúng ta ã h c cách s d ng chúng m t cách bao quát trong các tr ng h p và chúng ta s nghiên c u sâu vào chúng trong quy n sách này.

s d ng l p display list classes, Chúng ta c n ph i b t u v i DisplayObject class và d n xu ng nh hình 4-3 là th t các display list classes.

D i ây là m t s gi i thích ng n ng n cho các display list classes, chúng s làm cho vi c th o lu n c a chúng ta tr n ên rõ rang h n:

(Nh ng th mà thong ch a b t k 1 i t ng nào bên trong nó (khong ph i là **DisplayObjectContainer** ch ng h n trong sách h hay dung v i t “child” – mong các b n c n th n)

DisplayObject

B t c cái gì mà có th t n t i trong danh sách màn hình (díplay list) là m t i t ng màn hình, và nh ng l p chuyên d ng h n c d n xu tra t DisplayObject.

Shape

Cái này là m t hình ch nh t, hình ê-líp, c k , vân vân, t o rat vi c v nh ng công c .

V . Ban hoàn toàn có th tao ra 1 shape trong flash r t d dàng

Bitmap

Cái này là m t mã lệnh AS bitmap c t o ra khi th c hi n s d ng l p BitmapData. Chú ý r ng m t s file d nh d ng JPG khi import vào flash không th s d ng gi ng nh 1 ActionScript bitmap, nh ng khi t o ra m t hình d ng (shape). tuy nhiên, b n có th t m t JPG import nó cho màn hình.

Video

là m t i t ng màn hình vi êô, ít nh t c ng có th yêc u flash ch i m t vi êô, h n là m t thành ph n vi êô.

InteractiveObject

L p này bao g m b t k i t ng màn hình nào Ng i dùng Có th Interact v i vi c s d ng con chu t ho c bàn phím. Nó ch a c s d ng thao tác danh sách màn hình.

B n có th làm vi c v i nh ng i t ng bên trong nó (có lv th p h n)

BIÊN SO N L I T SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE J

SimpleButton

L p này c s d ng t o ra m t nút mà có t ng ng theo ch c n ng v i nh ng nút mà b n có l có bên trong giao di n authoring.

Tuy nhiên, trong ActionScript 3.0, b n có th bây gi t o ra nh ng nút trên (v) c n th n b i vi c s d ng nh ng i t ng màn hình khác i u ch nh Lên trên, qua, xu ng, và nh ng tr ng thái và nh ng va ch m.

TextField

L p này bao g m v n b n ng và nh p vào nh ng ph n t v n b n, có th ki m soát V i ActionScript.

DisplayObjectContainer

L p này t ng t nh DisplayObject bên trong mà nó tham chi u t i nh i u k i u i t ng màn hình. S khác nhau ây , tuy nhiên, là i t ng này có th ch a ng nh ng i t ng con (child). T t c các Container i t ng màn hình (DisplayObjectContainer) u là nh ng i t ng màn hình, ch nh ng i t ng màn hình mà có th có i t ng con bên trong (child) là nh ng Container i t ng màn hình. Cho ví d , m t video là m t i t ng màn hình, nh ng nó không th có con các i t ng bên trong (child). M t movie clip là M t i t ng màn hình, và nó có th có i t ng con bên trong (child), vì v y nó c ng là M t i t ng màn hình

DisplayObjectContainer

. Thông th ng, b n s làm vi c tr c ti p v i l p này khi vi c s d ng display list, tìm ki m children or ancestors (các i t ng tr c nó, bao b c i t ng ang xét). Thông th ng, b n s thao tác m t ho c h n nh i u i t ng khác nhau trong m t container.

Có b n lo i display object containers:

stage

Can` ph i Nh , Chính Stage là mot ph àn display list. cái này dieu` hanh` b t k t ng tác nào gi a các i t ng. Chính nó Là m t display object container.

Sprite

M i t i ActionScript 3.0, m t sprite n gi n là m t movie clip mà không có m t timeline.

Nhi u m à l nh thao tác ActionScript tiêu bi u th c hi n s d ng movie clips ch yêu c u m t khung (frame)

T o ra timeline c a movieclip. So, the size and administrative overhead of the timeline is unnecessary. As you become more accustomed to ActionScript 3.0, and begin to consider optimization more frequently, you may find yourself using sprites more than movie clips.

Loader

L p này c s d ng t i nh ng tài nguyên ho c cái jif ó dành cho cho danh sách màn hình, bao g m bitmaps và SWFs khác.

MovieClip

Nó là movie clip mà b n có l bi t và yêu thích no nh m t k t qu vi c t o ra chúng trong giao di n authoring, via ActionScript, Ho c C hai.

AVM1Movie

L p này làm vi c v i SWFs ã load vào flash b ng cách s d ng ActionScript 1.0 ho c 2.0. AVM1, (Mà ng cho Th c t ActionScript Làm b ng máy 1) c d tr cho SWFs mà s d ng ActionScript 1.0 và / ho c ActionScript 2.0, Trong khi AVM2 c s d ng cho SWFs mà s d ng ActionScript 3.0. Vì Ng i s d ng flash s d ng hai c s m à riêng bi t, nh ng máy th c t khac nhau này không ph i là thích h p v i nhau.

L p AVM1Movie cung c p m t cách thao tác nh ng thu c tính màn hình c a tài nguyên SWFs, Nh ng Không làm d dàng vi c truy n d li u gi a ActionScript 3.0 và SWFs c h n. ph i c hoàn thành b i nh ng ph ng ti n khác, gi ng nh LocalConnections. Chúng ta s th o l u n v ch c n ng này ch ng 13

MorphShape và StaticText

Hai l p này i di n m t hình d ng tween và ph n t v n b n t nh, t ng ng, tr c ti p có th ki m soát Via ActionScript. Tuy nhiên, chúng là b ph n c a nh ng l p màn hình b i vì chúng th a k nh ng thu c tính,

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE J

methods, Và Nh ng s ki n t l p cha m DisplayObject c a nó. Cái này làm cho nó là m t v trí quay m t ph n t v n b n t nh.

Displaying the Display List

(hi n th Display List)

Nó h u ích ôi khi, c bi t khi b n ang t o ra nhi u màn hình nhi u i t ng v i n hi u s ph c t p l ng vào nh ng i t ng-t c là bên trong i t ng ó ch a ng nhi u i t ng kh ác, tham chi u danh sách màn hình và phân tích n i dung c a Nó. H àm script này,b n s c tìm th y trong h s trace_display_list.fla

```
function showChildren(dispObj:DisplayObject):void {
```

```
    for (var i:int = 0; i< dispObj.numChildren; i++) {
```

```
        var obj:DisplayObject = dispObj.getChildAt(i);
```

```
        if (obj is DisplayObjectContainer) {
```

```
            trace(obj.name, obj);
```

```
            showChildren(obj);
```

```
        } else {
```

```
            trace(obj);
```

```
        }
```

```
}
```

```
showChildren(stage);
```

Trong VD này hàm s ghi ra toàn b nh ng children c a stage

dòng 1 n dòng 12 là nh ngh a c a hàm showChildren, dòng 13 là g i no ra.

T dòng 2 n dòng 10 là vòng l p **for**, vòng l p này s k t thúc chi thong còn m t “children” (ch c là m t lo i i t ng màn hình b t k nào ó) trên **Díplay object**. S vòng c 1 p leis xác nh b ng thu c tính numChildren (có ki u d li u s nguy ên) **Dòng 3 ch a giá tr c a c a m i “child”** (có l là s ít c a “children”) trong danh sách màn hình (d ísplay líst), m i child này c xác nh b ng ph ng pháp (Methods) getChildAt() **nh m Xác nh i t ng màn hình “child” thong qua nh ng l n l p vòng**

Khó hi u quá nh -1 y luôn cái VD o n m à trên gi i thích luôn: L n u tiên chi i = 0 ,“child” u tiên s c tr v ,trong l n l p th 2 – “child” th 2 c ng s c tr v t ng t .

Dòng 4 s ki m tra xem “child” chúng ta ang xét có ph i là 1 **DisplayObjectContainer** hay khong. **is** trong dòng th 4 c ng là 1 lo i toán t (gi ng nh +, -, *, / ...) N u i t ng là **DisplayObjectContainer** thì u tiên chúng ta s trace ra tên c a nó ã **obj.name** và sau ó là n i t ng ang xét (thu c lo i **DisplayObjectContainer**) là **obj**, sau ó s dùng chính hàm showChildren() tìm nh ng “child” con n m bên trong **DisplayObjectContainer** ó và ti p t c in ra các “child” con ó b ng l nh trace (d nhiên là có “child” n m trong i t ng là **DisplayObjectContainer** mà ta m i dùng câu l nh 4 kiêm tra), trong tr ng h p i t ng ban u (child) thong ph i là **DisplayObjectContainer** this m à l nh s nh y n câu l nh d òng 7-8 và c ng in ra tên c a no.

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER 'S GUIDE J

Dòng 9 là k t thúc c a c u trúc **if** và dòng 10 là k t thúc c a vòng l p **for** sau ó bi n I t ng them 1 d n v (i++) và vòng l p l p lei l n 2 cho n chi trên **màn hình chính** khong còn m t “child” nào.

L u ý: qua VD trên ta nh n th y r ng khi flash duy t mă 1 nh và a vào file swf cho chúng ta xem th k t qu thì t t co hàm, l nh, ... u ā c duy t (biên d ch) xong xuôi. Vì th m i có chuy n hàm **showChildren()** v n c s d ng trong ph n nh ngh a c a hàm **showChildren()** chi trong c u trúc c a hàm này thì quá trình nh ngh a v n ch a xong (mă 1 nh g i dòng 6 trong chi hàm nh ngh a k t thúc dòng 12).

The **showChildren()** function in action

(s d ng hàm **showChildren()**).

Mă 1 nh các b n th y trên chi là 1 ng s t v n n u chúng ta khong cung c p m t hay nhi u “child” tren **Stage** (có th là **DisplayObjectContainer** ho c là khong)

B n có th t o m t c u trúc “children” theo hình v sau minh h a cho mă 1 nh có th th c hi n.

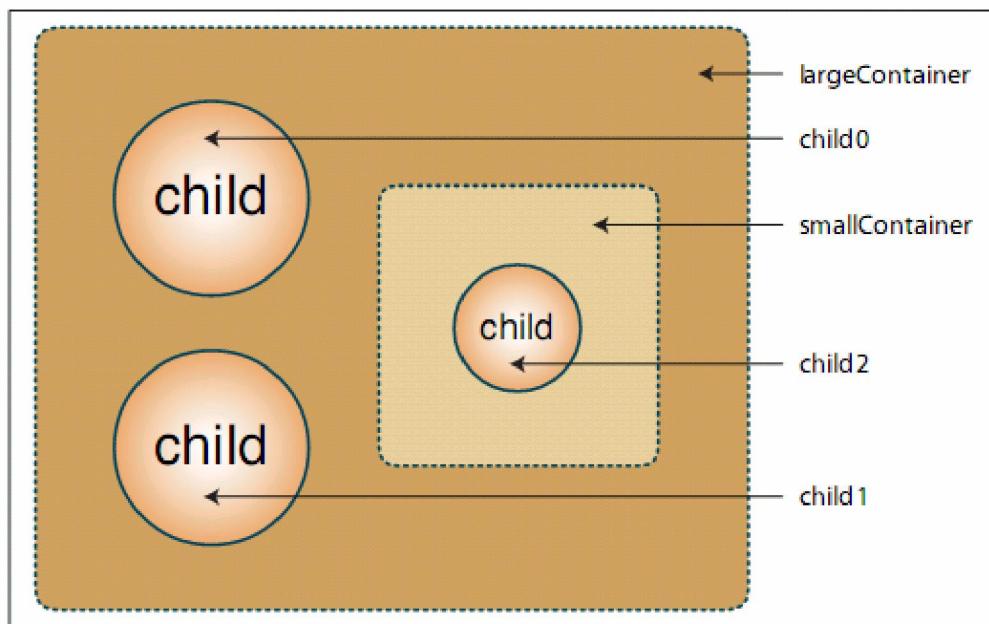


Figure 4-4. A look at the stage of trace_display_list.fla

Và ây là k t qu hi n ra c a s output – no cho ta th y toàn b nh ng i t ng ang có trong **Stage** (tên c a chúng) – nh ng i t ng con là “child” con c a các **DisplayObjectContainer**

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE

```
root1 [object MainTimeline]
largeContainer [object largeContainer_1]
[object Shape]
smallContainer [object smallContainer_2]
[object Shape]
child2 [object MovieClip]
[object Shape]
[object StaticText]
child0 [object MovieClip]
[object Shape]
[object StaticText]
child1 [object MovieClip]
[object Shape]
[object StaticText]
```

Cách s p x p này là không p m t – chúng ta có th coi ti n mă l nh ban u c a s output cho chúng ta m t k t qu d quan sát h n b ng cách them vào các mă l nh sau : nh ng mă l nh in m là nh ng mă c thêm vào:

```
function showChildren(dispObj:DisplayObject, indentLevel:Number): void {
    for (var i:int = 0; i < dispObj.numChildren; i++) {
        var obj:DisplayObject = dispObj.getChildAt(i);
        if (obj is DisplayObjectContainer) {
            trace(padIndent(indentLevel), obj.name, obj);
            showChildren(obj, indentLevel + 1);
        } else {
            trace(padIndent(indentLevel) + obj);
        }
    }
}
showChildren(stage, 0);
function padIndent(indents:int):String {
    var indent:String = "";
    for (var i:Number = 0; i < indents; i++) {
        indent += " ";
    }
    return indent;
}
```

Và chúng ta có KQ a ra nh sau:

Chúng ta s d ng them 1 bi n hàm
showChildren() là **indentLevel** bi n này
dung xác nh level c a i t ng chúng
ta ang xét thu c lv m y và s th t vào u
dòng t ng ng v i lv c a nó– n u lv = 0

B ND CH NÀY C VI TB I HUYETSAT
12/1 – THPT QU CH CHU - B ND CHL

```
root1 [object MainTimeline]
largeContainer [object largeContainer_1]
[object Shape]
smallContainer [object smallContainer_2]
[object Shape]
child2 [object MovieClip]
[object Shape]
[object StaticText]
child0 [object MovieClip]
[object Shape]
[object StaticText]
child1 [object MovieClip]
[object Shape]
[object StaticText]
```

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE

this hi n nhiên chúng ta s khong th t dòng (i t ng dang xét n m trên Stage)

S l n th t nh i u hay it ph thu c lv v i làn lv + 1 thi **indentLevel** +1, hàm **padIndent()** c nh ngh a d i s cho chúng ta bi t c n ph i th t u dòng bao nhiêu cho p (các b n t nghiên c u nha mă l nh ó d h u mà :D).

Adding and Removing Children

(them và xóa “children” – (m t cách t ng))

t o 1 MovieClip b ng cách ng (dung mă l nh) chúng ta s s d ng nhi u n l nh **new MovieClip()**, t t nhiên MovieClip t o b i cách này s là r ng (vì chúng ta ch a cho flash biet movieclip v a t o có nh ng cái ji bên trong no!). Chúng ta s h c cách t o 1 MC trong Library ph n sau.

VD: **var mc:MovieClip = new MovieClip();**

Mc b n v a t o là m t MovieClip chính hi u – nhung chúng ta c n them no vào trong danh sach màn hình b ng các s d ng ph ng pháp

addChild(mc);

Gi s r ng, MovieClip **mc** b n v a t o n m trong 1 MovieClip khác có tên navBar : **navBar.addChild(mc);**

chúng ta có th t o shape và sprite m t cách t ng t :

var sp:Sprite = new Sprite();

addChild(sp);

var sh:Shape = new Shape();

addChild(sh);

N u các b n làm quên v i AS 1 ho c 2 this ubi tr ng m i movieClip t o ki u này luôn có 1 thu c tính chi u sâu i kèm hang v i AS3 chúng ta thong quan tâm t i v n ó vì no ā c flash t o m t cách t ng. Th m chí chúng ta có th thay i thu c tính position c a MovieClip c t o hang v/d ó chúng ta s bàn ph n sau ch ng này.

Adding Symbol Instances to the Display List

Chúng ta có th t o 1 MovieClip trong c a s library b ng cách th c hi n b ng sau:

Chúng ta s h c cách v b ng mă l nh cho nh ng MC r ng chapter 8 – Còn hi n nay chúng ta chi t p trung vào nh ng i t ng trong library

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER 'S GUIDE J

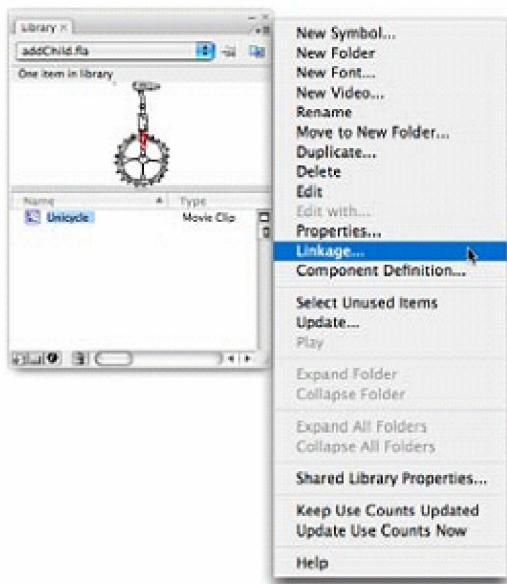


Figure 4-5. Accessing a symbol's linkage information

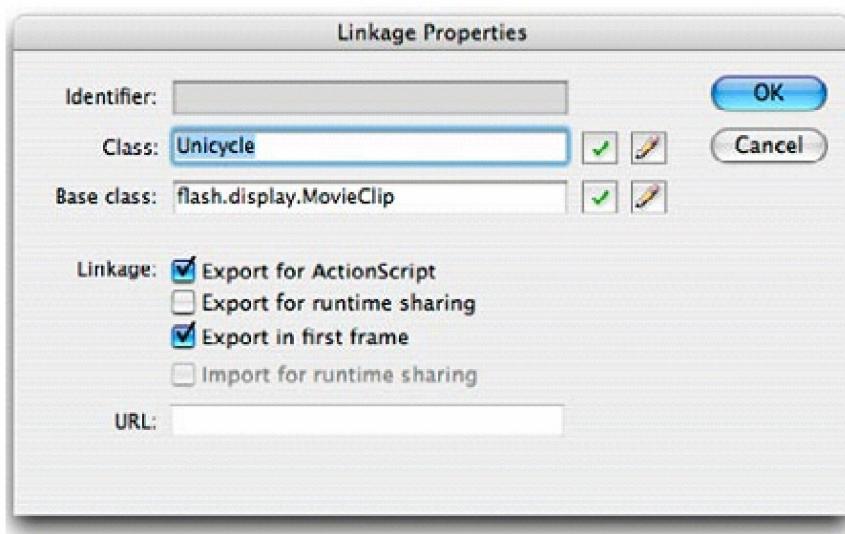


Figure 4-6. Entering a class name for a movie clip in the library Linkage Properties

```
var cycle:MovieClip = new Unicycle();
addChild(cycle);
```

Trong VD tren ban da tao ra 1 MC moi' co ten la` cycle them nguyen ban? San co' trong flash ma` lop' (class) rieng cua no' co' ten la` Unicycle.

Tu` bay gio` ban co the thao tac voi mc **cycle** nayf mot cach binh` thuong nhu thay doi vi tri cua no (mac dinh chi dua vao` thi` (x,y) = (0;0)):

Cycle.x = 100;

B N D CH NÀY C VI T B I HUYETSATHIENHA – thanh_vinh648@yahoo.com
12/1 – THPT QU C H C HU - B N D CH L U HÀNH TRÊN <http://WWW.VNFX.COM>

Cycle.y = 100;

Using addChildAt()

(su dung ham` addChildAt())

Su dung ham addChildAt() nay` chung ta se tao ra 1 Child co' thuoc tinh position lon' nhat so voi cac child da duoc tao ra truoc do' (dieu` do' co' nghia child nao` tao ra sau thi` luon dung' truoc cac child duoc tao ra truoc no) Dieu nay lam` viec dat item len tren tat ca cac item khac trong flash tro len that de dang`. Tuy nhien se huu ich hon neu child chung' ta dat co the o mot vi tri dac biet tren man` hinh` do chung ta thiet lap

```
var inc:uint = 0;  
stage.addEventListener(MouseEvent.CLICK, onClick,  
false, 0, true);  
function onClick(evt:MouseEvent):void {  
    var ball:MovieClip = new Ball();  
    ball.x = ball.y = 100 + inc * 10;  
    addChildAt(ball, 0);  
    inc++;  
}
```

Ma lenh tren kha' don gian? Dieu` moi' duy nhat la chi moi~ lan` click chuot (thuoc ve` Mouse Event) thi` mot uqa bong' moi' se duoc tao ra o mot vi tri nhat dinh. (chi bien inc = 0 è toa do (x;y) = (100;100) , chi inc = 1 è (x;y)=(110;110) ...

Tat nhien trong library chung ta can` 1 doi' tuong san co voi class **Ball** va nhung MovieClip moi' tao deu` co ten la **ball** co thuoc tinh position cung` bang 0 (nhung nhu da noi o tren tuy cung 1 vi tri (position) = 0 nhung em nao` ra sau thi` phai len truoc J).

Display Objects and References to Stage and Root

(Nhung doi tuong man` hinh` va` tham khao cach trinh` dien no' tren **stage** va` **root**)

Chung ta muon tao ra movieClip va` co the thay doi no neu muon – chung ta se thuc hien viec do' moi. Luc' va tat ca nhung Object khac cung` loai se thay doi theo (co the su dung chung ham` , ma lenh ...)

khi mot doi' tuong them vao` trong display list thi`no' se co nhung thuoc tinh cua **stage** va` **root** hop le (co the su dung)

BIÊN SO N L I T SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE ↴

Neu doi tuong khong phai 1 trong nhung ds man` hinh` thi` thuoc tinh cua **stage** se tra ve la null, va thuoc tinh **root** khi tra ve khi doi tuong la displayobjectcontainer chua mot child trong do', ta xet VD sau:

```
//tao mot danh sach man hinh moi'  
var mc:MovieClip = new MovieClip();  
// gia tri tra ve cua stage va root la null  
trace(mc.stage);  
trace(mc.root);  
//them object vao` display list  
addChild(mc);  
//bay gio` thuoc tinh stage co hieu luc nhung root thi` khong (no tra ve null)  
trace(mc.stage);  
trace(mc.root);
```

ma lenh co the khong hoat dong chinh xac neu ban khong thiet lap cho no' mot ke' hoach hoan` hao, xet vd sau, lap trinh` vien dang co gang thiet 1 p vi tri toa do cho mot **mc** vua` tao nhung anh ta se that bai vi` **mc** c t o ra trên n n c a stage , do ó, thu c tính **stage.stageWidth** không th do mc truy xu t c b ng m t thu c tính c a mc (**mc.stage.stageWidth/2**)

```
var mc:MovieClip = new MovieClip();
```

```
mc.x = mc.stage.stageWidth/2;
```

```
addChild(mc);
```

de su loi tren chung ta se lam cong viec rat don gian la thay i câu l nh 2 b ng cách g i tr c ti p thu c tính **stageWidth** c a stage

```
var mc:MovieClip = new MovieClip();
```

```
mc.x = stage.stageWidth/2;
```

```
addChild(mc);
```

khong ph i lúc nào chúng ta c ng s d ng **root** vì b n than **root** là m t i t ng màn hình gián ti p. b n luôn them i t ng vào danh sách màn hình chi t o ra chúng tránh nh ng k t qu b t th ng di n ra.

Removing Objects from the Display List and from Memory

(xóa b i t ng t danh sách màn hình và b nh)

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER 'S GUIDE J

Vì c t o và xóa i t ng trong flash là quan tr ng nh nhau

lo i b m t i t ng c thêm vào danh sach màn hình b ng mã l nh b n có th s d ng ph ng pháp (methods) **removeChild()**

VD: **removeChild(ball);**

xóa b m t i t ng màn hình m t lv (position) nh t nh chúng ta dùng **removeChildAt(lv);**

VD: chung ta se th c hi n ví d ng c v i VD c a **addChildAt()** à c th o lu n ph n tr c, s d ng vòng l p for t o nh ng qu banh trên lv 0 và l n l t xóa b chúng m i chi click chu t

for (var inc:uint = 0; inc < 20; inc++) {

var ball:MovieClip = new Ball();

ball.x = ball.y = 100 + inc * 10;

addChild(ball);

}

stage.addEventListener(MouseEvent.CLICK, onClick, false, 0, true);

function onClick(evt:MouseEvent):void {

removeChildAt(0);

}

Preventing out of bounds errors

(ng n ch n mä l nh có th n y sinh l i)

khi mä lnh trên à xóa b i t ng cu i cùng thi n u b n ti p t c click thông báo sau s hi n ra: "the supplied index is out of bounds." B i vì b n ang c g ng remove m t child ra kh i v trí (lv) 0 mà th c t thong còn child nào c , chúng ta có th s a l i này b ng cách thêm mä lnh sau:

function onClick(evt:MouseEvent):void {

if (numChildren > 0) {

removeChildAt(0);

}

}

numChildren là s child còn trên ch ng trình.

Removing objects from memory

(xoá bo m t i t ng kh i b nh)

Chung ta da thao luan ve` event listener trong chuong 3, dieu` quan trong can` nho' la` flash co the khong co du ket qua tra ve` ma` chi thao tac tren bo nho'. Neu doi' tuong ban dang su dung khong con` huu dung nua, ban co the xoa' bo no' khoi bo nho' nham tiet kiem them nhieu` tai nguyen cho may tinh. Thong thuong chung ta hay loai bo doi' tuong khoi danh sach man` hinh` nhung chung van con` ton` tai o RAM ca khi doi' tuong khong duoc tham gia vao` chuong trinh`. Vd sau la su don gian hoa' cua vd truoc day chung ta se loai bo doi' tuong tu` danh sach man hinh` va tu bo nho' RAM.

var ball:MovieClip = new Ball();

ball.x = ball.y = 100;

addChild(ball);

stage.addEventListener(MouseEvent.CLICK, onClick, false, 0, true);

function onClick(evt:MouseEvent):void {

this.removeChild(ball);

//ball da bien khoi danh sach man` hinh` xong van con` nam lai trong RAM

trace(ball)

ball = null;

//ball gio` day da ra di vinh vien

BIÊN SO N L I T SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE

```

trace(ball)
stage.removeEventListener(MouseEvent.CLICK, onClick);
}

```

Chú ý tham so cuoi cung cua ham addEventListener() – da dc trinh bay o chuong 3.

Dong` lenh 15 la can` thiet vi` no dam bao cho dong` lenh 5 khong phat sinh loi, (van co the chay duoc neu dong` 5 phat sinh loi – chi ma` stage khong con` 1 doi tuong nao` do bi xoa` het – dong` 15 la mot ban sao luu hoan hao cua dong lenh 5, khong phai la su loai bo nhung listeners khong can thiet)

As an added review of best practices, line 15 emphasizes the concept of removing event listeners covered in Chapter 3.

This is a good example of this practice, since using a weak reference in the last parameter of the addEventListener() method in line 5 is not sufficient. This is because a reference to the stage will always exist. Remember, weak references are a best practice backup plan, not a substitute for explicitly removing your unwanted listeners. For additional information, please review Chapter 3

Managing Object Names, Positions, and Data Types

(quan ly ten doi' tuong, vi tri (position) va kieu du lieu)

Flash la mot cong cu do` hoa 2D nhung van co khai niem ve` truc Z trong he truc toa do Oxyz. Tuy nhien truc Z chi nham` bieu dien xem doi' tuong nao se duoc dung truoc doi tuong khac, no khong co nghia bieu dien ve` mot hinh` chieu phoi canh 3D nhu cua cac cong cu do` hoa 3D khac.

Finding Children by Position and by Name

Chung ta co the tim` “children” trong danh sach man hinh bang ten hoac vi tri (position) cua no

S d ng phu ng pháp **getChildAt()**, b n c o th làm vi c v i child u tiên (c a) m t container s d ng cú pháp quen thu c : **var do:DisplayObject = getChildAt(0);**

do se la doi tuong tham chieu truc tiep den doi tuong o vi tri 0 (nh ng doi' tuong do thuc chat chi la 1) Neu ban khong nho ro lam` ve` vi tri cua doi tuong dang can tham chieu ma` chi biet ten cua no la circle thi hay su dung cu phap sau:

var do:DisplayObject = getChildByName("circle");

do se la doi tuong tham chieu truc tiep den doi tuong co ten la circle (nh ng doi' tuong do thuc chat chi la 1)

cu i cùng, n u b n c n bi t s nh v (v trí) c a m t i t ng màn hình trong màn hình li t kê, nh ng b n ch có tên c a nó, thì b n c o th s d ng ph ng pháp **getChildIndex()** hoàn thành m c ích c a b n
VD

var do:DisplayObject = getChildByName("circle");

var doIndex:int = getChildIndex(do);

Casting a Display Object

(th vai m t i t ng màn hình)

Chung ta van thuong thao luan ve doi tuong man hinh nh kieu du lieu chu' khong noi rieng ve` MovieClip – cac ban khong nen nham` lan ve` 2 khai niem nay` boi vi ngoai MovieClip thi mot child van co the la sprite, shape ...

Flash khong yeu cau` ve` khai bao kieu du lieu – nhu khi tao ra doi tuong bang cach dong (ma lenh)

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE J

De noi cho Flash nghe 1 cau hoi chi b n ang trong m t movieclip, ban c n ph i tham chi u n b m c a no, t c là timeline chính c a flash, mã lệnh sau a flash n frame 20 ang t mäl nh trong MovieClip:

parent.gotoAndStop(20);

Nhung flash khong biet rang gotoAndStop la mot phuong phap co the su dung duoc cua display object container (hien tai dang la **stage**) (di den frame 20) nen ban se gap loi sau:

Call to a possibly undefined method gotoAndStop through a reference with static type flash.display:DisplayObjectContainer

Ban co the sua lai ma lenh nhu sau:

MovieClip([ten mc].parent).gotoAndStop(20);

Ma lenh tren cho biet parent (cha me) cua doi' tuong dang dat ma lenh chinh la **stage** va no thuoc kieu **MovieClip** flash se hieu **stage** co the su dung phuong phap **gotoAndStop()** va` ma lenh se hoat dong tot'

Xem VD:

Tao o frame 10 mot key frame moi', ve mot hinh` ji` do' lam` dau'

var mc:MovieClip = new MovieClip();

addChild(mc);

stage.addEventListener(MouseEvent.CLICK, onClick, false, 0, true);

function onClick(evt:MouseEvent):void {

//gap loi unrecognized method of display object container neu go doan ma sau:

//mc.parent.gotoAndStop(10);

//dua den (main timeline) as a movie clip lam cho MovieClip ho tro lenh **gotoAndStop()**

MovieClip(mc.parent).gotoAndStop(10);

}

Changing the Display List Hierarchy

(Thay i s Phân c p Danh sách Màn hình)

Voi AS3 chung ta co the thay doi thuoc tinh position (do sau) cua mot doi; tuong trong flash mot cach de dang (dieu` nay khac biet so voi cac phien ban truoc cua AS) Dieu nay giup chung ta quan ly cac doi' tuong de dang hon.

Depth Management

(quan ly thuoc tinh do sau)

Them mot child vao` trong danh sach man` hinh`, flash khong yeu cau` chung ta cung cap thong tin ve` thuoc tinh do sau cua doi' tuong child vi` dieu` nay` mac dinh se duoc flash thuc hien giup ban mot cach tu dong, ban da su dung 2 ham` sau: **addChild()** va **addChildAt()**, ban van co the su dung phuong phap **addChild()** cho mot doi tuong dang ton` tai trong danh sach man` hinh`, phuong phap **addChild()** se xoa bo doi' tuong cu va tao lai doi' tuong no voi mot thuoc tinh do sau moi – doi' tuong moi tao se duoc the hien tren cung` (co do sau cao nhat trong lv cua no).

var mc1:MovieClip = new MovieClip();

mc1.name = "clip1";

addChild(mc1);

var mc2:MovieClip = new MovieClip();

B ND CH NÀY C VI TB I HUYETSATHIENHA – thanh_vinh648@yahoo.com

12/1 – THPT QU CH CHU - B ND CH L U HÀNH TRÊN <http://WWW.VNFX.COM>

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE

```
mc2.name = "clip2";
addChild(mc2);
trace(getChildAt(0).name);
trace(getChildAt(1).name);
```

gia tri cho ta o cua so output la :

clip1

clip2

Chung ta co the giao thich nhu sau: tu` dong` 1 à 6 la` ma lenh de tao 2 MovieClip, “clip1” duoc tao truoc va` no nhan mot muc do sau la` 0 (khi do ta moi dung` cau lenh **trace(getChildAt(0).name)**; de goi thuoc tinh **name** cua no’, tuong tu cho “clip2”.

Tuy nhien ban muon di chuyen do sau cua mc1 tu` 0 len 1 va` mc2 se nhay xuong do sau la 0, chung ta co the them doan ma sau va xem ket qua:

```
addChild(mc1);
trace(getChildAt(0).name);
trace(getChildAt(1).name);
```

Co 3 cach de thay doi, them bot' thuoc tinh do sau theo truc Z trong flash , mot trong so chung la su dung phuong phap swapChildren() dung de hoan doi vi tri giua 2 doi tuong, duoc dat o vi tri tham so cua ham`

VD:

swapChildren(mc1, mc2);

se trao’ doi vi tri cua 2 movieclip mc1 va` mc2. Khong can` quan tam xem tren truc Z mc1 cach mc2 bao nhieu

Trong truong hop ban chua de` cap ra doi tuong “child” ban co the su dung phuong phap **getChildByName()** boi vi` no cho phep chung ta tim` ra doi tuong “child” theo ten cua no – sau do’ hay thiet lap thuoc tinh do sau cua no bang phuong phap **swapChildrenAt()** . Can luu y la phuong phap nay se trao doi toan bo nhung “child” thuoc 2 do sau bat ky`:

VD:

swapChildrenAt(0, 1);

// se trao doi toan` bo nhung child thuoc do sau 0 cho toan bo nhung child thuoc do sau 1

Cuoi cung` , Ban co the chi dinh mot chi so moi cho toan bo nhung Object display. Doan ma dung de automatically bring any

display object that is rolled over with the mouse to the top of the visual stacking order.

```
this.addEventListener(MouseEvent.MOUSE_OVER, onBringToTop, false, 0,
true);
```

```
function onBringToTop(evt:MouseEvent):void {
```

```
    this.setChildIndex(evt.target, this.numChildren-1);
```

```
}
```

This script accomplishes its task by setting the child’s display list index to the highest level possible. The script first determines the number of children in the display object container (in this case, the main timeline) and then, because ActionScript arrays are zero-based (meaning the first item is item 0), it subtracts 1 to get the highest existing index in the display list. For example, if there are three items in the display list, their indices would be 0, 1, and 2.

The number of children would be 3, and 3 minus 1 equals 2—the highest index in the list. Figure 4-7 illustrates.

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE ↴

Xem VD sau:

Tao 4 MovieClip co ten folder0,folder1,folder2,folder3

Them doan ma nay` vao timeline chinh'

```
this.addEventListener(MouseEvent.MOUSE_OVER, onBringToTop, false, 0, true);
```

```
function onBringToTop(evt:MouseEvent):void {  
    var folder:MovieClip = evt.target as MovieClip;  
    setChildIndex(folder, numChildren - 1);  
    trace(getChildAt(0).name);  
    trace(getChildAt(1).name);  
    trace(getChildAt(2).name);  
    trace(getChildAt(3).name);  
}  
}  
}
```

numChildren cho biet so child tren stage.

setChildIndex(ten mc, chi so do sau) la ham` thiet lap thuoc tinh do sau cua child, luu y la **chi so do sau** chi nam trong gioi han cua do sau lon' nhat hien co trong chuong trinh`

Theo VD tren lan luot folder1,folder2,folder3,folder4 co vi tri 0,1,2,3 (do cach dat vao` stage)

Chi re chuot den folder1 chung ta duocKQ:

folder0

folder2

folder3

folder1

Chi re chuot den folder2 chung ta duocKQ

folder0

folder3

folder1

folder2

Chi re chuot den folder3 chung ta duocKQ

folder0

folder1

folder2

folder3

Chi re chuot den folder0 chung ta duocKQ

folder1

folder2

folder3

folder0

Reparenting Children

Mot nhanh vu dat ra la chung ta co the chuyen mot child tu` 1 parent nay` sang parent khac hay khong?

Mot parent (cha me) chua' 1 hay nhieu child (dua' tre) va` viec chung ta quan tam la dua tre se duoc chuyen tu` parent nay` sang parent kia nhu the nao`.

Vd chung ta Nghien cuu se la viec di chuyen “moon”(child) tu` bau` troi`(parent) sang bau`troi` kia (parent)

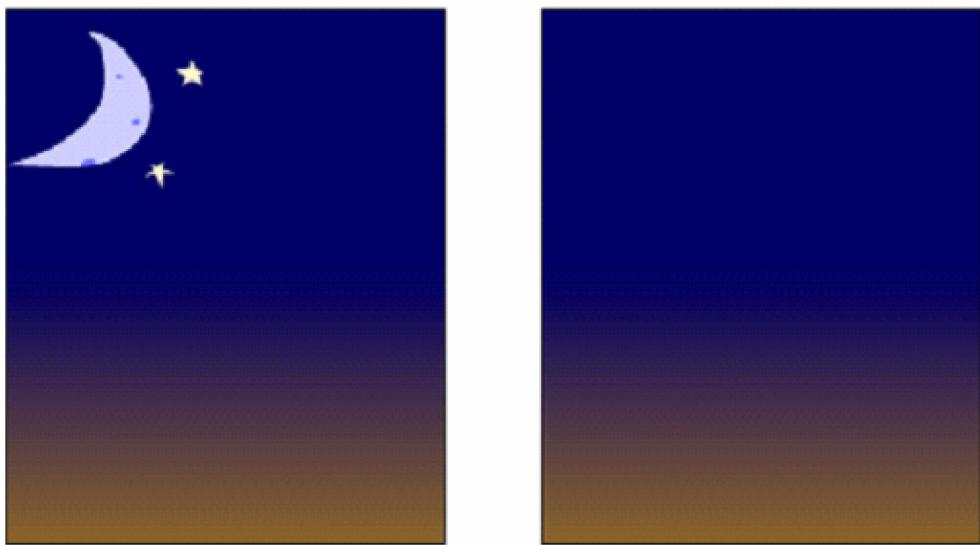


Figure 4-8. In reparenting.fla, the moon and stars become a child of the clicked sky

```
sky1.addChild(moon);
this.addEventListener(MouseEvent.MOUSE_DOWN, onDrag, false, 0, true);
this.addEventListener(MouseEvent.MOUSE_UP, onDrop, false, 0, true);
Doan ma nay` them child moon vao` sky1 va` them cho no 1 vai` event listener cho vien keo tha (this o
day hieu la man` hinh` chinh', do do` sky1 (va them ca sky2 se duoc bo sung sau) se co the su dung 2
ham` onDrag va` onDrop trong MouseEvent.MOUSE_DOWN, MouseEvent.MOUSE_UP.
Ham` nay` se them moon vao trong sky Chi duoc click. Hanh dong nay se xoa bo tu chinh parent truoc
do` and them no chi click vao doi tuong.
```

Ngoai` vien reparenting (thay doi cha me) the moon. Ham` con` cho phep vien keo doi tuong dang duoc
click

```
function onDrag(evt:MouseEvent):void {
    if (evt.target == moon) return;
    evt.target.addChild(moon);
    evt.target.startDrag();
}
```

Bo sung ham` sau day de ket thuc vien keo tha:

```
function onDrop(evt:MouseEvent):void {
    evt.target.stopDrag();
}
```

Ham **addChild()** rat huu dung vi` no vua` xoa moon cu va` them moon moi` vao` parent moi`.

A Dynamic Navigation Bar (thanh nut chon dong)



Figure 4-9. A dynamically generated navigation bar

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE J

Chung ta se tao 1 thanh nut chon gom` 5 nut nhu tren, cua so output se hien ra ten cua tung` nut chi chung ta ckick vao nut do

Bat dau Du an’ cua chung ta bang ma lenh sau day:

```
var btnNum:uint = 5;
var spacing:Number = 10;
var navBar:Sprite = new Sprite();
// khai bao doi tuong navBar
addChild(navBar);
// dua navBar vao stage
```

Dong` 1 va 2 khoi tao nhung gia tri : so nut cua thanh va` khoang cach cac nut trong thanh (10 pixels)

Dong` 3 tao 1 container de chua cac nut lenh ma chung ta se tao o dong` lenh 5

// trong navBar chung ta tao nhung nut lenh theo vong lap **for**

```
var btn:SimpleButton;
for (var i:Number = 0; i<btnNum; i++) {
    btn = new Btn();
    btn.x = spacing + i*(btn.width+spacing);
    btn.y += 5;
    btn.name = "button" + i
    btn.addEventListener(MouseEvent.CLICK, onTraceName, false, 0, true);
    navBar.addChild(btn);
}
```

//tao nen` cho navBar, tai do sau la 0 de cho cac nut lenh co the nam truoc nen` (movieClip **bg**)

```
var bg:MovieClip = new NavBarBack();
bg.width = btnNum*(btn.width + spacing);
bg.width += spacing
navBar.addChildAt(bg, 0);
```

navBar.x = (navBar.stage.stageWidth - navBar.width)/2;

navBar.y = 20;

// thiet lap vi tri cua navBar tren man hinh cua flash

function onTraceName(evt:MouseEvent):void {

trace(evt.target.name);

}

// ham nay de ec – khoi giao thich :D

evt.target nham chi den doi tuong dang duoc noi den trong ham` (doi tuong dang thuc thi ham`)

Sau nay ban se hoc cach ve mot Doi tuong hoan chinh bang AS khong can su dung den tai` nguyen san co trong cua so library

TypeError: Error #1009: Cannot access a property or method of a null object reference.

at Ball\$init()

at ball_fla::MainTimeline/ball_fla::frame1()

CHAPTER 5 TimeLine Control

1. Playhead Movement

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER 'S GUIDE J

(S di chuyển playhead)

M trong nh ng k n ng L p trình AS c b n nh t là kh n ng nh h ng trong nh ng o n phim (Movies) trong flash. B n s th ng dùng k n ng này i u ki n s phát i ho c chayj o n phim c a MovieClip bên trong o n phim chính. Mã lệnh trong ch ng này r t n gi n và b n s có th t t o ra nh ng file d án khác cho riêng mình. b t u chúng ta s th o lu n v vi c ch y (play) ch y o n phim (playback) d ng lai (stop) và nh y sang m t khung hình (frame) khác.

D án u tiên mà chúng ta quan tâm s là t o ra m t h s có y 4 nút l nh c b n i u ki n o n phim trong flash: ó là **play, stop, gotoAndStop, gotoAndPlay**. Chúng ta b t u nào:

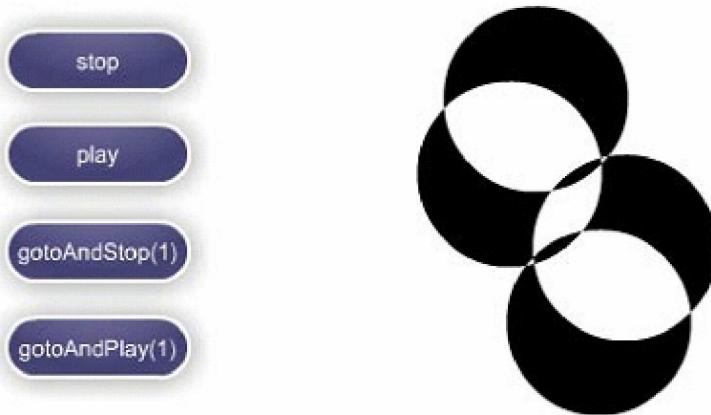


Figure 5-1. navigation_01.fla demonstrates simple navigation.

S d ng nh ng công c ho t hình nào ó t o ra m t o n ho t hình trong flash nh nh n F5 hay F6 t o cho no nhi u frame nhá, chuy n nó lên m t layer m i và hãy ng t tên cho nh ng frame mà hãy tin t ng vào các s th t c a khung hình (khung hình th nh t s là s 1 ... cái này thì ai c ng bi t roài).

u tiên là ho t hình truy n th ng, ngh a là ho t hình s i t frame 1 à frame Cu i cùng, khi ch y no có th d ng lai và ch y ti p, Chúng ta s c n 2 nút **playBtn** và **playBtn** th c hi n công vi c này.

Sau ó b n không mu n ho t hình truy n th ng mà s nh y n m t frame mong mu n, b n c n s d ng AS th c hi n vi c này, ó là lý do chúng ta thêm vào h s 2 nút **gotoPlayBtn** và **gotoStopBtn**

T o 4 button nh hinh và thêm o n mă sau:

```
// nh h ng n gi n
stopBtn.addEventListener(MouseEvent.CLICK, onStopClick, false, 0, true);
playBtn.addEventListener(MouseEvent.CLICK, onPlayClick, false, 0, true);
gotoPlayBtn.addEventListener(MouseEvent.CLICK, onGotoPlayClick, false, 0, true);
gotoStopBtn.addEventListener(MouseEvent.CLICK, onGotoStopClick, false, 0, true);
//Mă l nh goij c a 4 nút
function onStopClick(evt:MouseEvent):void {
```

```
stop();
}
function onPlayClick(evt:MouseEvent):void {
    play();
}
function onGotoPlayClick(evt:MouseEvent):void {
    gotoAndPlay(1);
}
function onGotoStopClick(evt:MouseEvent):void {
    gotoAndStop(1);
}
```

//các hàm c nh ngh a

Mã lệnh trên g n vào timeline chính cu flash lên no s tác ng n o n phim chính c a flash. M à c th là ho t hình mà chúng ta v a t o. Ngh a là **stop** t c là o n phim chính c a flash d ng lai, **Play** t c là o n phim chính c a flash ang ch y.

Chúng ta tham kh o h s navigator_01.fla xem k t qu và m t s thêm vài mã l nh trong h s navigator_02.fla cho chúng ta thêm thông tin t c a s output K.

Frame Labels

(Nhãn khung hình)

Labels là tên nhãn c a m t o n frame nào ó, o n này c t p h p trong m t key frame và d nhiên Labels c ng s là tên mà Chúng ta t cho key frame ó.

Chúng ta có th t tên cho khung hình ô **frame** trong c a s **properties**, chúng ta s thao tác v i khung hình thông qua tên ā t. Tao m t MovieClip, trong MC có tên pages trong MC ó t o 3 key frame t tên chúng l n l t là page1, page2, page 3, kéo nó vào Màn hình chính, sau ó t o 3 button có tên one, t o, three, o n m à sau c thêm vào trong màn hình chính (timeline chính) c a flash

```
//D ng vi c ch y o n phim chính (t c là cái file swf lun)
pages.stop();
```

//button event listeners

```
one.addEventListener(MouseEvent.CLICK, onOneClick, false, 0, true);
two.addEventListener(MouseEvent.CLICK, onTwoClick, false, 0, true);
three.addEventListener(MouseEvent.CLICK, onThreeClick, false, 0, true);
```

// nh h ng tên frame t t h n thông qua s khung c a MovieClip pages

```
function onOneClick(evt:MouseEvent):void {
```

```
    pages.gotoAndStop("page1");
}
```

```
function onTwoClick(evt:MouseEvent):void {
```

```
    pages.gotoAndStop("page2");
}
```

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE

```
function onThreeClick(evt:MouseEvent):void {  
    pages.gotoAndStop("page3");  
}
```

Mã lệnh AS có thể cho chúng ta hình dung trước KQ, hãy thử xóa vài frame trong 1 nhóm key frame nào đó và bấm thử để xem xét Frame Labels và cho chúng ta thấy điều gì đang diễn ra frame trong movie.

New Timeline ActionScript

Chúng ta hãy cách tạo ra TimeLine mới cho flash, các bank có thể tham khảo ở file **frameLabels_02.fla** dưới đây:

```
onFrameStart();
{
    trace(scenes[0].name);
    trace(scenes[0].numFrames);
    trace(scenes[0].labels[0].name);
    trace(scenes[0].labels[0].frame);
    // ghi ra các thông tin của một phim chính
    // đang movie clip pages trên scene "primary"
    stop();
}
```

```
trace(scenes[0].name);
trace(scenes[0].numFrames);
trace(scenes[0].labels[0].name);
trace(scenes[0].labels[0].frame);
// ghi ra các thông tin của một phim chính
// đang movie clip pages trên scene "primary"
pages.stop();

//button event listeners
onePlus.addEventListener(MouseEvent.CLICK, onOnePlusClick, false, 0, true);
output.addEventListener(MouseEvent.CLICK, onOutputClick, false, 0, true);
labelCheck.addEventListener(MouseEvent.CLICK, onLabelCheckClick, false, 0, true);
```

```
//Nhận ý định frame để chuyển sang một Frame Labels ("page1")
function onOnePlusClick(evt:MouseEvent):void {
    pages.gotoAndStop(getFrame("page1", pages) + 4);
}
```

```
//ghi ra thông tin của movie, scene, label, and frame information
function onOutputClick(evt:MouseEvent):void {
```

```
    trace("on phim chính có " + scenes.length + " scenes.");
    trace("scene hiện tại là " + currentScene.name + ".");
    trace("nó có " + currentScene.numFrames + " frame(s).");
    trace("và " + currentScene.labels.length + " label(s).");
    trace("tên nhãn đầu tiên của scene thứ 2 là " + scenes[1].labels[0].name + ",");
```

BIÊN SO N L IT SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE

```
trace("nó n m " + scenes[1].labels[0].frame + ".");
trace("Movie clip 'pages' có " + pages.currentLabels.length + " labels.");
trace("nhãn cu i cùng c a nó là " + pages.currentLabels.pop().name + ".");
}

//Ki m tra xem m t tên khung có t n t i hay không
function onLabelCheckClick(evt:MouseEvent):void {
    trace(isFrameLabel("page3", pages));
}

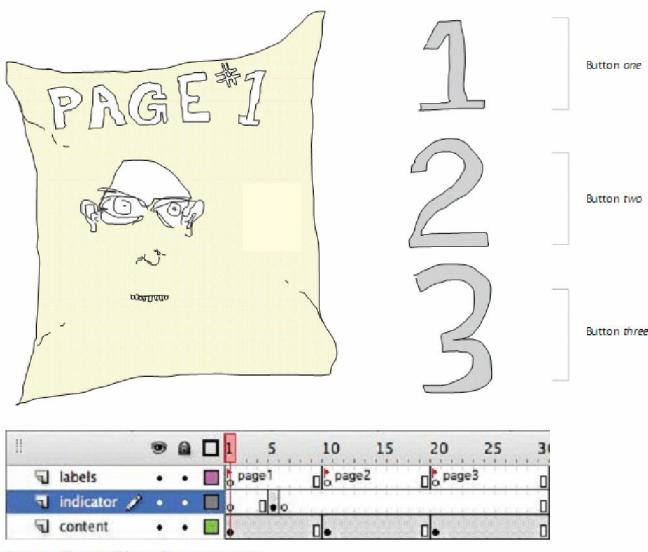
//Ch p nh n tên c a movieClip và tên khung c a nó s      c tr v b ng s c a khung      c
// ánh s th t trong flash
function getFrame(frLabel:String, mc:MovieClip):int {
//Khai báo các biến trong hàm, ki u d li u c a hàm.
    for (var i:int = 0; i<mc.currentLabels.length; i++) {
//vòng l p for s ch y t frame u tiên [0] n frame cu i cùng c a t t c labels
        if (mc.currentLabels[i].name == frLabel) {
            return mc.currentLabels[i].frame;
        }
    }
    return -1;
}

//không tr v giá tr n u nh chung ta không tìm th y giá tr nào c i u này có nghĩa Mc
//chúng ta // ang ch y không có frame nào c ( ang tr ng)
}

//Ham này tr v k t qu true n u tìm th y tên label có t n t i
function isFrameLabel(frLabel:String, mc:Object):Boolean {
    for (var i:int = 0; i<mc.currentLabels.length; i++) {
        if (mc.currentLabels[i].name == frLabel) {
            return true;
        }
    }
    return false;
}

// Giá tr false tr v khi câu lệnh if không      c th a mǎn.
```

Sau khi hoàn thành h s c a chúng ta có thể th này: b n có thể tham khảo file m u xem KQ:



Bạn có thể xem trên hình frameRate.fla. chúng ta thấy:

Frame Rate

Khi bạn sử dụng AS3 bạn mới có thể thay đổi cách Frame Rate mà cách nhanh và thời gian.

Giá trị mặc định là 12 và bạn có thể tăng lên hoặc giảm xuống.

Nếu bạn chỉnh frame rate mà thì hiển nhiên tất cả các thành phần trên màn hình chính flash sẽ thay đổi theo cách mà này và cách sau sẽ giúp bạn điều chỉnh.

bị tắc tia quang, mà hình ảnh sẽ cho

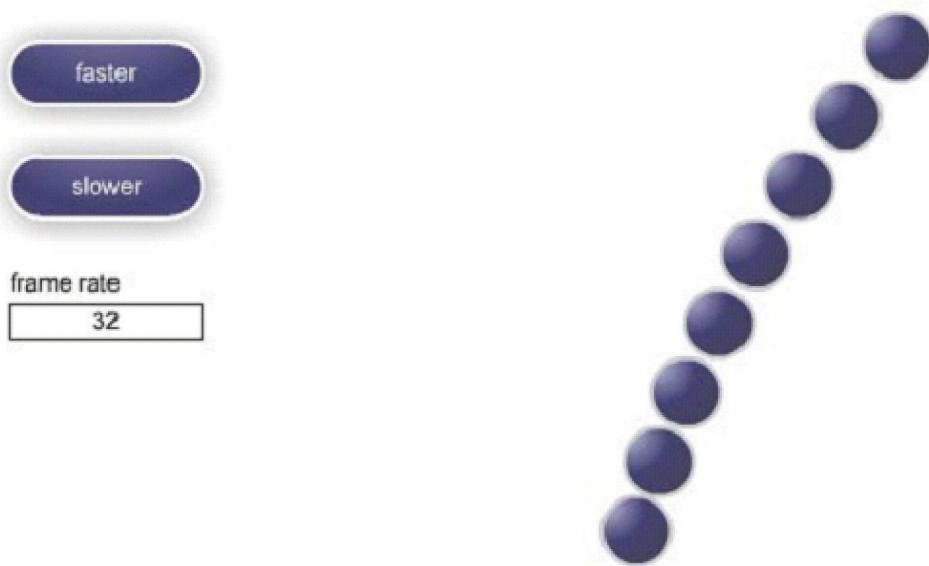


Figure 5-4. frame_rate.fla with buttons on the left that increase and decrease the frame rate, which control the speed of the animation on the right

Chúng ta có thể tạo ra một vài hoạt hình trên màn hình của flash và mã lệnh wocj thêm vào timeline chính, frame đầu tiên.

Mã lệnh chúng ta nghiên cứu trước đây:

info.text = stage.frameRate;

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE

```
//info.text      c liên kết với im ô và n b n ng bên ngoài màn hình chính
faster.addEventListener(MouseEvent.CLICK, onFasterClick, false, 0,
true);
//mã lệnh gán giá trị 1 button có tên faster
slower.addEventListener(MouseEvent.CLICK, onSlowerClick, false, 0,
true);
// mã lệnh gán giá trị 1 button có tên slower
//các event listener
function onFasterClick(evt:MouseEvent):void {
    stage.frameRate += 5;
    info.text = stage.frameRate;
}
//hàm onFasterClick sẽ tăng frame rate lên 5 nhanh (tức là tăng thêm 5 khung
//còn //chỉ trong 1 giây
function onSlowerClick(evt:MouseEvent):void {
    if (stage.frameRate > 5) {
        stage.frameRate -= 5;
    }
    info.text = stage.frameRate;
}
```

A Simple Site or Application Structure

(cấu trúc của một trang web đơn giản)

Chúng ta có một trang hoàn chỉnh là demosite.fla các bộ phận có thể tham khảo nó bao gồm phông chữ và các hình ảnh và lập trình AS3 và flash, Nhìn vào trang chúng ta sẽ không sâu mà chúng ta sẽ Nghiên cứu và mã lệnh là chủ yếu.

//gán trang cho nextSection

```
var nextSection:String = "";
```

//thêm button event listeners

```
section1.addEventListener(MouseEvent.CLICK, navigate, false, 0, true);
section2.addEventListener(MouseEvent.CLICK, navigate, false, 0, true);
section3.addEventListener(MouseEvent.CLICK, navigate, false, 0, true);
```

//nếu nhấp vào một nút sẽ

```
function navigate(evt:MouseEvent):void {
```

BIÊN SO N L IT SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE J

```
//Chuyển giá trị nextSection thành tên của nút bấm mà bạn đang bấm vào nó  
nextSection = evt.target.name;  
play();  
}
```

iều này có ý nghĩa là trên phim chính câu lệnh **gotoAndPlay(nextSection);** sẽ làm cho trên phim chính chuyển label mà có tên nút bấm quy định
VD nút **section1** sẽ chuyển trên phim chính label **section1...**
(các bước thực hiện nằm trên mã trên trong file màu).

Undocumented: Adding Frame Scripts to Movie Clips at Runtime

Phím này nói về pháp mc.addFrameScript(), thêm mã lệnh vào một frame nào đó, nháy đúch vì nó có thể ngưng ngay lập tức sau khi mc chuyển sang frame 39.

```
function onStopMC() {  
    mc.stop();  
    trace(mc.currentFrame);  
}
```

mc.addFrameScript(mc.totalFrames-1, onStopMC);

//hàm này sẽ gán mã lệnh vào frame số thứ 39 (từ frame 0 đến 39) của MovieClip mc theo cách // ánh sán a flash (tức là frame 40)sau khi mc chuyển sang frame 39 thì hàm **onStopMC()** // sẽ被执行 và mc sẽ dừng lại

up nguyên vẹn lên cho các bạn xem:

To finish off our discussion of timelines, we want to show you an undocumented method for adding frame scripts to movie clips at runtime. As always, be careful using undocumented ActionScript, testing your implementation thoroughly and trying not to rely on its use for final production, if possible. In addition to making no warranties as to current reliability, there's no guarantee that future versions of Flash Player will support the technique.

To implement this feature, you need to create a movie clip with at least 10 frames, and no other ActionScript in the file. The sample file addFrameScript.fla demonstrates the feature. The method we will use is

```
<movieclip>.addFrameScript(<framenum>,<function>,
...rest);
```

Adding the method to a movie clip instance name, you can dictate that any function be called when the specified frame number is reached. The ellipsis followed by “rest” indicates this function will accept an unlimited number of comma-delimited arguments. In this case, the structure requires pairs of frame number, function, frame number, function, and so on. In the following example, only one frame script is added.

First, a function is defined to trace the word “ten” followed by a space and then the actual current frame of the movie clip (see Chapter 2 for more information about the this keyword).

```
function frameTen() {
    trace("ten " + mc.currentFrame);
}
```

```
mc.addFrameScript(9,frameTen);
```

Then the addFrameScript() method is used, specifying that the frameTen function be added to a frame indicated by the number 9. This, however, is not frame 9, as this parameter relies on the fact that ActionScript is a zero-based array language.

Therefore, the first item is 0, the second item is 1, and so on.

So, the 9 in this syntax specifies frame 10. (Again, for more information, see Chapter 2.)

When you run this file, the movie clip will animate and, when it reaches frame 10, will trace the following to the Output window:

```
ten 10
```

This is a significantly simplified example, but there could be many uses for this feature. One popular use could be adding a stop() action to movie clips. It is relatively trivial to stop a movie clip on its first frame by using

```
<instance>.stop()
```

as we did with the pages movie clip in this chapter. However, it requires a little more effort and is usually processor-intensive, such as a combination of a repeating event and a conditional to stop a movie clip on its last frame.

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER 'S GUIDE

However, with this unsupported method, you could use something like the following:

```
function stopMC() {  
    mc.stop();  
}  
mc.addFrameScript(mc.totalFrames-1, stopMC);
```

This instruction would add a script to the last frame of the movie clip (remember, this parameter is based on calling the first frame zero), which would then be executed only when that frame is reached.

Don't forget to use this unsupported method with caution, if at all, in any production files. Elsewhere, have fun and experiment!

CHAPTER 6 - OOP

Object-oriented programming

(l p trình h ng i t ng)

OOP c hi u nh là m t v n v gi i quy t k thu t, chia mā lēnh c a b n thành nh ng ph n nh , làm mā lēnh tr lēn n gi n (chia thành nh ng i t ng), làm cho các d án và ng d ng c a chúng ta tr lēn d qu n lý h n.

Nh ng i t ng chúng ta lập trình c n càng chi ti t càng t t b i vì chúng ta có th s d ng chúng v i các thành ph n i t ng khác trong flash mà hōk c n ph i vi t l i mā l nh.

Vì c l a ch n OOP l p trình là m t vi c r t hi n nhiên khi l p trình nh ng d án l n hay khi b n ph i làm vi c theo nhóm v i cùng 1 nhóm l p trình viên,tuy nhiên có th ch p nh n OOP nh m t chi n l c phát tri n h n.

however, adopting OOP as a development strategy can be less obvious, and even debated.

M c ích c a ch ng nh m cho b n la nh ng cái nhìn khát quát nh t v OOP, chu n b cho nh ng d án l n h n c a b n, b n hñy s d ng các file m u theo dõi chapter này

D án l n h n s có n nt ng OOP nh ng c ng s ch a ng nh ng bài t p mà b n t o ra s d ng k thu t th t c, a b n t i c mô hình c a tác gi cu n sách.

Vì c khi nào l a ch n OOP s ph thu c vào vi c hi u nh ng l i ích c a chúng, (OOP):

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE J

Class: Là một lớp trong Object-Oriented Programming (OOP) và là một phần của các hàm, biến và mảng có giá trị trả về. Chúng ta sẽ xem xét cách sử dụng chúng sau, chúng ta sẽ nêu rõ trong chapter 1.

Inheritance: Là một khái niệm trong Object-Oriented Programming (OOP), nó cho phép bạn thêm những特性的 khác vào một đặc tính đã tồn tại. **set without reinventing the wheel, as the saying goes**

iều này có nghĩa là có thể mở rộng một lớp hiện có trên một lớp cũ để hoàn toàn và phân chia tinh thần, giúp bạn tiết kiệm thời gian và cải thiện các đặc điểm.

Composition: Là một cách kết hợp các lớp khác nhau trong Flash, do đó một lớp thành phần thay thế, nó có khả năng thu hút sự chú ý của người dùng và biến.

Encapsulation: Là một khái niệm không thường được sử dụng cho một lớp khác nhau, mà là một lớp riêng lẻ, chúng ta có thể có một lớp riêng lẻ để giữ bên ngoài, cho phép chúng ta kết hợp các phần của một lớp và chúng ta sẽ nhìn thấy cấu trúc của lớp mà không cần đến các lớp khác.

Polymorphism: Là một tính chất, cho phép nhiều lớp khác nhau có thể thực hiện cùng một hành động. Các lớp có cùng tên không cùng nhau, nhưng chúng ta có thể thực hiện cùng một hành động bằng cách gọi là `method` có cùng tên nhau. Điều này giúp chúng ta dễ dàng mở rộng và học cách sử dụng chúng. Lập trình object-oriented và có thể sử dụng những phương pháp của lớp chính và trả về kết quả cho lớp mà không cần phải gọi.

OOP không phải là một khái niệm mới, tuy nhiên nó có ích trong nhiều ứng dụng, lập trình hướng đối tượng có cấu trúc thích hợp với các chương trình nhỏ và ngắn. OOP, mặc dù có thể phức tạp, nhưng nó chỉ là một cách tiếp cận khác.

Tuy nhiên trong chương này chúng ta vẫn quan tâm đến OOP để biết là các ví dụ, cách sử dụng, và lý thuyết về OOP mà không phải là trình OOP khi không cần thiết, mà chỉ là một cách tiếp cận.

Classes

(1 p)

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE J

Trong Ch 1 chúng ta th o lu n v 3 mô hình l p trình chung nh t: liên t c, th t c và h ng i t ng (mình ch d ch ph n class trong l p trình h ng i t ng tho i J)

Chúng ta nói r ng, l p trình th t c là m t b c t phá m i so v i l p trình liên t c vì nó không b h n ch v s n i ti p trong mã lệnh theo chi u t trên xu ng d i mà chúng ta có th nhóm các mã l nh có nhiệm v chung trong 1 th t c (ho c là hàm trong AS)

L p mà n n t ng c a OOP và có 1 b n ã làm vi c v i chúng m t luc nào ó, dù b n m i chi tham gia vào l p trình nh ng b n, b n ã h c n ch ng 6 này ngh a là b n ã t ng ti p xúc v i l p mà có th ch a th c thi nó b i vì khi b n ch y ch ng trình thì nhi u th ng sau **scenes** mà flash ch y ang s d ng l p riêng c a nó hoàn thành công vi c c a b n.

Chúng ta ã nói n cách làm sao thao tác các s ki n **EventDispatcher** ã nói n chapter 3, bao g m nhi u khái ni m khác nhau nh DisplayObjectContainer, DisplayObject, Và Nhi u con cháu h hàng khác c a nó.

B n có com th y chúng ta ã h c quá nhi u không khi i n ch ng g ng l p trình oop, không t t c chi là i m b t u c a chúng ta, t t co nh ng VD chúng ta ã xét có liên quan n l p tr c ây chi nh m giúp minh h a v cách s d ng l p, Cùng nhìn l i mă l nh b n có th i u khi n Ch ng trình thông qua m t MovieClip, b n có th thi t l p nhi u thu c tính **x, y, rotation, alpha, và nhi n h n th** nh ng ph ng pháp Start(),Stop(),goto... và co s ki n **Event.ENTER_FRAME** c ng chi là m t l p trong Movieclip, b n có th t o ra l p v i var mc:MovieClip = new MovieClip(); ngh a là b n ang t o ra l l p MovieClip m i trên n n c a l p MovieClip c , b n có th ch a có nhi u kinh nghi m trong vi t l p hang hấy tích l y kinh nghi m s d ng chúng m t cách nhu n nhuy n và chính xác

Custom Class Review

(t ng quát c a m t l p ph bi n)

Ch ng 1 trình bày v cách vi t l l p n gi n, chúng ta có th xem l i ph n ó ho c có m t cái nhìn và rút ra kinh nghi m qua o n mă sau ây:

package {

```
import flash.display.MovieClip;
```

```
public class Main extends MovieClip {
```

```
public function Main() {  
    trace("Flash");  
}  
}
```

Cu trúc **package {}** này là bút bù c cho tất cả các lớp, hãy t file lớp (.as) t i thi m c cùng với chương trình hiện hành ho c xem ph n **Classpaths** bi t cách t file as nh ng th m c khác nhau.

Cu trúc của lớp c khai báo t dòng 5 n dòng 11 nó c m r ng c a i t ng s n có là movieclip trong flash có nghĩa là t c các sкин, nh ng ph ng pháp, và nh ng thu c tính của lớp MovieClip s s n sàng t i l p **Main** này, và ó là lý do t i sao là lớp MovieClip c n nh p vào trong lớp **Main**. cái này th hi n "s th a k" c s d ng trong dòng 1 nh 7 à 9, nó là cu trúc của lớp **Main**. L p có th c s d ng m t cách t ng ho c chúng ta thêm câu lệnh sau: **var main:Main = new Main();** khai báo lớp cho flash

B n có th t o ra nhi us th hi n c a l p trong flash n u nh mā lệnh cho phép. Tóm l i b n áh c các thao tác c b n có th s d ng **custom classes** (nh ng l p thông th ng).

Classpaths

(ng d n n l p)

B n có ít s l a ch n khi thao tác v i l p, b i l chúng ta m i th o lu n v 1 l p t trong th m c cùng file v i swf, chúng ta có th t các lớp trong các th m c khác nhau quán lý chúng d dàng h n, chúng ta c n ph i nh p nh ng l p này b ng mā l nh riêng, m t khác, chúng ta có th nh p các lớp s n có khác c a flash vào ch ng trình, k thu t này c ng c th c hi n m t cách t ng t cách nh p m t class t m t th m c nào ó:

D i ây là m t s VD:

```
import flash.display.MovieClip;
```

```
import myapp.effects.Water;
```

```
import flash.events.Event;
```

Khi vi t l p m t th m c nào ó b n c n ph i vi t **Classpaths** trong cu trúc **package {}** Nh VD sau:

```
package myapp.effects {
```

```
public class Water {  
    public function Water() {  
    }  
}  
}
```

Bây giờ th m mục **myapp** là th m c me c a th m c **effects** ch a file **Water.as** hay nói cách khác thì ây chính là 1 p c a b n, **public class Water {}** c u trúc này cho bi t nó s th a h ng các c tính c a l p **Water** ā c nh ngh a tr c ó (d nhiên t t c chi là VD-vì hi n th i chúng ta không có h s nào v VD ā nêu c :D)

B n có th qu n lý, xây d ng các l p gi ng nh trong th vi n c a flash, b n s không mu n nhân ôi l p ó nhi u l n trong quá trình s d ng mà mu n l p ó s g n theo file fla c a b n

Thay vào ó b n có th cho flash bi t v ng ān n class c a b n là **classpath** và flash s t hi u ph i làm gì, VD nh flash t hi u khi b n ch cho nó én class thông th ng nào ó mà b n t o flash s t ng nh p thêm 1 p **flash.display**. th c hi n, u tiên các b n s ch **classpath** cho ph n **ActionScript section** a b ng h i tho i **Preferences** trong flash, 1 a ch n phiên b n AS ang dùng là 2 hay 3

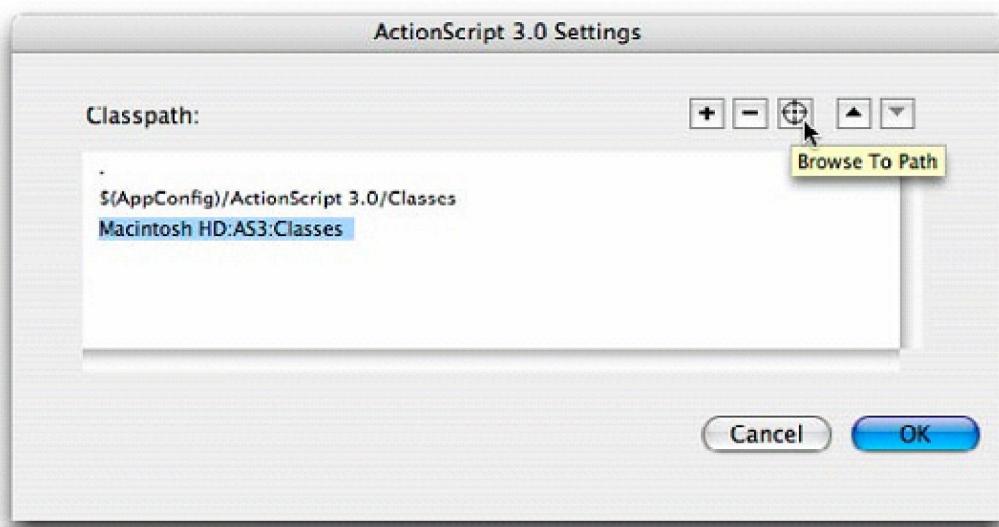


Figure 6-1. Adding your own classpath to Flash's ActionScript preferences

Sau khi th c hi n công vi c này, sau này flash s t tìm ki m các th m c mà b n ā ch nh và b n ā có m t th vi n riêng cho các l p.

Inheritance

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER 'S GUIDE J (s th a k)

Mã 1 nh sau là m t cách th hi n ý t ng c a s thi t k
Mã 1 nh s t o ra m t l p m r ng có tên là **Box** và nó c m r ng t l p c a MovieClip, k t q a là nó có th truy c p n t t c các thu c tính, ch c n ng và các l p khác nhau c a MovieClip nh thu c tính t a X (dòng 22) và 1 p h a c a nó (dòng 13) nh v m t ng màu xanh tao thành m t cái h t có 4 c nh màu xanh, chúng ta s th o lu n v cách v b ng vect s d ng mã 1 nh ch ng 8, VD này n gi n là ta b t u t bút v iêm (0,0) v n (100,100) và kick th c ng v là 1 pixel và màu v c quy nh theo mã màu (có th xem nh theo mã màu c a RGB)

```
1 package {
2
3     import flash.display.MovieClip;
4     import flash.display.Graphics;
5     import flash.events.Event;
6
7     public class Box extends MovieClip {
8
9         public var color:uint = 0x000099;
10
11        public function Box() {
12            //draw a shape at runtime
13            this.graphics.lineStyle(1, 0x000000);
14            this.graphics.beginFill(color);
15            this.graphics.drawRect(0, 0, 100, 100);
16            this.graphics.endFill();
17
18            this.addEventListener(Event.ENTER_FRAME, onLoop, false,
19            0, true);
20
21        public function onLoop(evt:Event):void {
22            this.x += 5;
23        }
24    }
25 }
26 }
```

Mã 1 nh trên cho chúng ta m t l p v m t i t ng mà khi g i thì i t ng ó s có hình d ng mà chúng ta ã miêu t trên, nó chuy n ng qua ph i 5 pixel 1 l n ch y qua m t frame do lệnh this.x += 5; c t trong hàm onloop(); tên hàm này không quan tr ng nh ng cái quan tr ng là nó t vào trong EventListener (ng i nghe s ki n) c a Event.ENTER_FRAME, chính nó s làm cho mã 1 nh c g i

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE J
m i khi flash duy t qua m t frame m i hay b t u l p l i t frame u tiên t o thành 1 vòng m i.

Symbol Base Classes

(l p c s c a các bi u t ng)

B n ā th y c cái hay c a tính k th a trong vi c truy su t k i t ng mà nó ā th a k , trong ch ng 4 b n ā làm i u này h n 1 l n trong khi ang s d ng các i t ng trong library b ng cách thêm nó vào ch ng trình chính c a flash b ng mā 1 nh. Bây gi t m t i t ng s n có trong flash, b n có th vi t m t l p cho nó nó có th k th a và phát huy nhi u y u t riêng c a nó mà b n c n, VD nh i t ng Square s n có b ng cách ban t o nó trong library, b n có th thêm l p sau cho nó:

```
package {  
    import flash.display.MovieClip;  
    import flash.events.Event;  
    public class Square extends MovieClip {  
        public function Square() {  
            this.addEventListener(Event.ENTER_FRAME, onLoop, false,  
                0, true);  
        }  
        private function onLoop(evt:Event):void {  
            this.x += 5;  
        }  
    }  
}
```

Mã lệnh trên c ng nh c a mā 1 nh chung ta ā th o lu n ph n tr c ch có i u là nó ā m t b o n mā v lēn hình d ng c a MovieClip Square

A More Traditional Look at Inheritance

(nh i u tính truy n th ng nhìn nh m t s k th a)

B n ā có ý t ng c b n v vi c làm sao m t l p ph thông (thông th ng) l i có th th a k nh ng thu c tính c a l p MovieClip s n có trong flash, chúng ta s mô t s th a k s m h n bàn lu n làm sao m t child có th th a k thu c tính c a parent (cha m) c a nó, t ng t chúng ta có các ví d nh l p các con chó k th a t l p ng v t, l p oto k th a t l p xe c ...

- A Dog class might inherit from an Animal class, a Ball class might inherit from a Toy class, and a Car class might inherit from a Vehicle class.

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE J

ó là nh ng vd c i n cho các l p trình OOP

L i u m t xe c là m t ô tô ho c m t xe t i — Ho c Th m chí m t máy bay ho c m t con thuy n, c cho cái ó quan tr ng - nó v n ch là m t xe c và chia s nhi u (thu c tính trong khi l u thông cùng) v i nh ng xe c khác.

Whether a vehicle is a car or a truck—or even a plane or a boat, for that matter—it is still a vehicle and shares much in common with other vehicles.

Chúng ta thi t l p các l p xe c d a trên nh ng nét chung c a nó – t c là t t c xe c u ph i có c i m này, VD nh nhiên li u, xe nào mà ch t n x ng (tr xe i n) Sau ó b n hñ y ngh n các ph ng pháp có th liên quan n thu c tính c a “nhiên li u” nh là m t cái xe s h t x ng trong bao lâu khi ang ch y ?—và chúng ta s cung c p x ng cho xe ch y – b n có th gi m s x ng hao phí n u b n có m t l ch trình du l ch t t và i u ó là k t qu cu i cùng mà chúng ta h ng t i.

Vehicle class

(l p xe c)

Chúng ta s xem xét m t d án m i v nh ng chi u xe khi ang ch y và t t nhiên là n o s t n x ng :D

Hãy t o m t l p c b n và t tên cho nó là *Vehicle.as* n gi n vì chúng ta ang xét n l p xe c , file này l u cùng th m c v i h s flac a b n, l p này s t o ra xe c , i u khi n cho nó ch y và thông tin t c a s output s chúng ta bi t chúng ta á i c bao nhiêu d m và s bình gas (*gallons of gas*) còn l i là bao nhiêu. K t qu là b n s bi t cái xe chúng ta ang ch y có th ch y c bao nhiêu d m cho n khi n h t gas (x ng)

L p chúng ta s có 4 thu c tính chung c a nó, bao g m ng á i qua b ng cách tiêu t n gas, nhiên li u gas còn có, s d m i c và thu c tính _go theo ki u boolean nh m cho bi t khi nào thì xe s ch y và d ng, n o s th hi n khi tr ng thái enable và không ho t ng khi tr ng thái *disable*

Nh ng thu c tính và ph ng pháp trong l p này thu c lo i chung nêncó th s d ng công c ng (t c là các l p khác có th th y chúng)

Chú ý n hàm onloop() (cái này á gi i thích roài)

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER 'S GUIDE J

```
1 //Vehicle.as
2 package {
3
4     import flash.display.MovieClip;
5     import flash.events.Event;
6
7     public class Vehicle extends MovieClip {
8
9         public var _gasMileage:Number;
10        public var _fuelAvailable:Number;
11        public var _milesTraveled:Number = 0;
12        public var _go:Boolean;
13
14        public function Vehicle(mpg:Number=21, fuel:Number=18.5) {
15            _gasMileage = mpg;
16            _fuelAvailable = fuel;
17            this.addEventListener(Event.ENTER_FRAME, onLoop, false,
18            0, true);
19    }
```

Hàm onloop() c t trong s ki n ENTER_FRAME cho chúng ta th y ây là hàm s ch y ch y u trong ch ng trình chính, nó ph i m b o các y u t sau: Khi thu c tính _go là úng (true) thì xe s b t u ch y và s d m mà xe ch y c s t ng t 0 n ... khi mà xe h t gas, thu c tính này t ng theo giá tr c a thu c tính “ng ã i qua b ng cách tiêu t n gas” (hi n nhiên r i), giá tr thu c tính “s l ng x ng còn” ph i gi m và khi nó < 1 thì thu c tính _go nh n giá tr false và xe d ng l i, c a s output cho chúng ta bi t chung ta ã i h t bao nhiêu d m v i m t1 ng gas nh t nh trên xe.

D i ây là chi ti t c a hàm onloop():

Trong hàm go() l nh **_go = true;** s giúp cho câu l nh if hàm onloop ho t ng, cái này gi ng nh vi c m và t t máy c a xe, chúng ta có th s d ng nhi u h th ng ph c t p h n nh ng t t nh t là gi cho VD càng n gi n càng t t.

```
19     public function onLoop(evt:Event):void {
20         if (_go) {
21             _fuelAvailable--;
22             _milesTraveled += _gasMileage;
23             if (_fuelAvailable < 1) {
24                 this.removeEventListener(Event.ENTER_FRAME,
25                 onLoop);
26             }
27             trace(this, _milesTraveled, _fuelAvailable);
28             this.x = _milesTraveled;
29         }
30     }
31     public function go():void {
32         _go = true;
```

```
33 }  
34 }  
35 }  
36 }
```

Main Flash file

(mã 1 nh file flash chính)

T i sao các b n không nh n Ctrl + Enter xem i u gì x y ra? :D

Mình ùa y vì chúng ta v a vi t xong m t class thui, m t class không có ý nghĩa j i n u chúng ta không t nó vào m t h s nào ó và ó là công vi c c a chúng ta trong ph n này. Tham kh o k t qu khi ch y ch ng trình:

```
//output  
[object Vehicle] 21 17  
[object Vehicle] 42 16  
[object Vehicle] 63 15  
...  
[object Vehicle] 336 2  
[object Vehicle] 357 1  
[object Vehicle] 378 0
```

Mã 1 nh thêm vào main flash (ch ng trình chính, nh v trí t c a nó trên timeline) r t n gi n nh ng tr c khi xem mã 1 nh, b n **không** ph i t tên c a class vào m c instance (tên phiên b n c a l p trong tr òng h p này là *Vehicle*) i u này không c phép vì flash s b nh m1 n gi a l p dành cho các i t ng s n có trong flash và các l p thêm vào ch y v i m t i t ng c ch nh trên ch ng trình chính, và mã lệnh d i ây s báo l i. N u b n làm theo l i mình nói thì mã 1 nh d i s không sao

var vehicle:Vehicle = new Vehicle(21, 18);

//l p vehicle s uyocj truy n 2 bi n _gasMileage nh n giá tr 21 và _fuelAvailable nh n giá tr 18 vì chúng ta khai báo chúng theo ki u **public**

addChild(vehicle);

//Khai báo và thêm cái xe vào màn hình

vehicle.go();

// kick ho t hàm go() c a l p *Vehicle* và xe s ch y

//d nhiên là chúng ta ch a ó cái xe nào (à v àu nó ch là movieClip r ng ch a mă

//l nh theo l l p có s n thoai) do ó hấy t ng t ng là xe ang ch y nh bay :D

Th t là hay n u chúng ta t o ra nhi u lo i xe khác nhau tham gia giao thông. Nh chúng ta thêm oto và xe t i vào ch ng trình, c hai lo i này u là l p m r ng c a l p xe c t t nhiên nó k th a m i thu c tính c ng nh ph ng pháp c a l p xe c .

BIÊN SO N L I T SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE

n g i n chúng ta s t hêm vào cách phân lo i m t ph ng pháp i u khi n t ng xe m t b ng m t hàm ph - m t sunroof Cho ô tô và m t tailgate Cho xe t i.

L p Car

//Car.as

package {

```
import flash.display.MovieClip;
import flash.events.Event;
public class Car extends Vehicle {
    public function Car(mpg:Number, fuel:Number) {
        _gasMileage = mpg;
        _fuelAvailable = fuel;
    }
    public function openSunroof() {
        trace(this, "opened sunroof");
    }
}
```

L p Truck (xe t i)

//Truck.as

package {

```
import flash.display.MovieClip;
import flash.events.Event;
public class Truck extends Vehicle {
    public function Truck(mpg:Number, fuel:Number) {
        _gasMileage = mpg;
        _fuelAvailable = fuel;
    }
    public function lowerTailgate() {
        trace(this, "lowered tailgate");
    }
}
```

Xem l i main Flash file

Chúng ta thêm 2 chi c xe vào ch ng trình c a flash, do ó mã l nh ch ng trình chính c n s a l i phù h p:

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE

//host .fla file

```
var compact:Car = new Car(21, 18);
compact.x = 10;
compact.y = 10;
addChild(compact);
//thêm xe car vào màn hình, v trí (10;10)
compact.openSunroof();
// ch y hàm Sunroof();
var pickup:Truck = new Truck(16, 23);
pickup.x = 10;
pickup.y = 100;
addChild(pickup);
//thêm xe t i vào màn hình, v trí (10;100)
pickup.lowerTailgate();
// ch y hàm lowerTailgate();
stage.addEventListener(MouseEvent.CLICK, onClick, false, 0, true);
function onClick(evt:MouseEvent):void {
```

```
    compact.go();
    pickup.go();
}
```

// khi click chu t thì 2 xe cùng ch y

Tr c khi b n click chu t 2 xe cùng ch y b n s th y c a s output hi n ra

```
[object Car] opened sunroof
[object Truck] lowered tailgate
```

Cái này dùng phân bi t 2 xe. 2 xe có nhiên li u và quãng ng ch y khi tiêu t n 1 n v gas khác nhau, do ó chuy n ng c ng nh quãng ng i c c a chúng khác nhau.

Composition

(S H p thành)

M c d u s th a k là m t cách r t chung trong l p trình h ng i t ng, nh ng nó không ph i là cách duy nh t cho nh ng l p có th làm vi c cùng nhau. **Composition** trong nhi u tr ng h p c ng r th u d ng. nó s biên so n l i t m t i t ng khác h n là th a k t m t i t ng khác cách t t nh t mô t s h p thành là tra 2 câu h i “nó là” ho c “nó có”. VD b n mu n thêm l p xe vào d a trên y nguyên l p c a xe c , n u v y thì câu h i là “nó là” và b n mu n biên so n l i l p l p xe d a trên l p xe c thì câu h i c a b n s là “nó có”. Chi c xe t i có th y nguyên l p c a

B ND CH NÀY C VI TB I HUYETSATHIENHA – thanh_vinh648@yahoo.com
12/1 – THPT QU CH CHU - B ND CH L U HÀNH TRÊN <http://WWW.VNFX.COM>

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE J
xe c , do ó s k th a trong VD này ho t ng t t nh ng n u l p xe c ng th a
h ng y nguyên l p c a xe c thì i u này l i không úng. Oto c m r ng t xe
c thì b n không th thay i c ji t xe c c , cho dù ó là l p xe nh ng n u chi c
xe c a b n có m t l p l p xe, ph tùng lung tung... thì ban l i có th d dàng thay i
l p xe a cho tài x khác

Chúng ta s thêm s **h p thành** vào trong mã l nh c a **Vehicle.as** b ng cách thêm
vào ó thu c tính c a l p bánh xe **_tires** , trong mã l nh ã có s n thu c tính cho oto
và xe t i.

//Vehicle.as

```
package {  
    import flash.display.MovieClip;  
    import flash.events.Event;  
    public class Vehicle extends MovieClip {  
        public var _gasMileage:Number;  
        public var _fuelAvailable:Number;  
        public var _milesTraveled:Number = 0;  
        public var _go:Boolean;  
        public var _tires:Tires;  
        public function Vehicle(mpg:Number=21, fuel:Number=18.5) {  
            _gasMileage = mpg;  
            _fuelAvailable = fuel;  
            this.addEventListener(Event.ENTER_FRAME, onLoop, false,  
            0, true);  
        }  
        public function onLoop(evt:Event):void {  
            if (_go) {  
                _fuelAvailable--;  
                _milesTraveled += _gasMileage;  
                if (_fuelAvailable < 1) {  
                    this.removeEventListener(Event.ENTER_FRAME,  
                    onLoop);  
                }  
                trace(this, _milesTraveled, _fuelAvailable);  
                this.x = _milesTraveled;  
            }  
        }  
    }  
}
```

```
public function go():void {  
    _go = true;  
}  
}  
}
```

L p trình vi ên r t m t m i ch phân bi t thu c tinh tires c a l p **Vehicle.as** v à tìm cách có th can thi p vào thu c tinh này trong các l p kh ác k th a c a l p **Vehicle.as**

Chúng ta v n xét 2 l p c a xe t i và xe oto

Car class

```
1 //Car.as  
2 package {  
3  
4     import flash.display.MovieClip;  
5     import flash.events.Event;  
6  
7     public class Car extends Vehicle {  
8  
9         public function Car(mpg:Number, fuel:Number) {  
10             _gasMileage = mpg;  
11             _fuelAvailable = fuel;  
12             _tires = new Tires("highperformance");  
13             trace(this + " has " + _tires.type + " tires");  
14         }  
15  
16         public function openSunroof() {  
17             trace(this, "opened sunroof");  
18         }  
19     }  
20 }
```

Chú ý thu c tinh tires ang mang giá tr g ì

Truck class

```
1 //Truck.as
2 package {
3
4     import flash.display.MovieClip;
5     import flash.events.Event;
6
7     public class Truck extends Vehicle {
8
9         public function Truck(mpg:Number, fuel:Number) {
10            _gasMileage = mpg;
11            _fuelAvailable = fuel;
12            _tires = new Tires("snow");
13            trace(this + " has " + _tires.type + " tires");
14        }
15
16        public function lowerTailgate() {
17            trace(this, "lowered tailgate");
18        }
19    }
20 }
```

Chú ý thu c tính `_tires` ang mang giá tr gì

New Tires class

(l p m i c a l p xe)

Chúng ta t o ra m t l p m i cho thu c tính `_tires` và m t hay nh i u l p ki u nà h p l i s b sung hoàn h o cho l p `Vehicle.as` ây chính là bi u hi n c a s h p thành mà u i m là chúng ta d d àng thay i thu c tính `_tires` ch khôn gò bó vào l p `Vehicle.as` nh các thu c tính khác.

```
//Tires.as
package {
    public class Tires {
        public var _type:String;
        public function Tires(type:String) {
            // óng vai trong s thay i ho t ng d a vào ki u l p xe
        switch (type) {
            case "snow":
                _type = "storm-ready snow";
        //ki u l p xe i vào tuy t
                break;
        }
    }
}
```

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER 'S GUIDE
case "highperformance":

```
    _type = "high-performance radial";
//ki u l p xe i trên ng cao t c.
    break;
default:
    _type = "economical bias-ply";
}
}

public function get type():String {
    return _type;
}
}

}
```

Main Flash file thì mã 1 nh không ph i thay i ji tuy nhiên kq output s khác:

```
[object Car] has high-performance radial tires
[object Car] opened sunroof
[object Truck] has storm-ready snow tires
[object Truck] lowered tailgate
[object Car] 21 17
[object Truck] 16 22
[object Car] 42 16
[object Truck] 32 21
...
```

Encapsulation

(ồng gói)

u tiên chúng ta có 2 khái ni m c n chú ý:

L p chính (**superclass**): c hi u là l p s cho m t l p khác th a k thu c tính c ng nh ph ng pháp c a nó.

VD trong ph n tr c, l p **MovieClip** là m t **superclass** vì nó cho l p **vehicle** th a k

L p ph , l p b sung (**subclass**): c hi u là l p s th a k t m t l p khác, nó có nh ng thu c tính c ng nh ph ng pháp c a **superclass**

VD trong ph n tr c, l p **vehicle** là m t **subclass** vì nh n th a k t l p **MovieClip** là m t **superclass**.

Trong t t c các vd trr c ây, chúng ta u khai báo nh ng bi n c a l p b ng **public** Khai báo ki u này r t ti n l i vì các o n mã ngoài l p (nh mã trong flash) s có th nhìn th y và can thi t vào các thu c tính c ng nh ph ng pháp c a l p, tuy nhiên

BIÊN SO N L IT SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE

ieuf này ng ngh a v i vi c m t s o n mā có th thay i giá tr c a l p m t cách vô tính ho c c ý b i ng d ng mà chúng ta có th nh n ra ho c là khōng và óng gói là m t bi n pháp h u hi u ng n i u này x y ra. n gi n là óng gói cho phép b n n nh ng thu c tính ho c ph ng pháp c a l p m t vùng d án nào ó c a l p nh ng v n cho phép b n thao tác v i chúng m t cách có ki m soát.

CHúng ta thay i **public** b ng **private** i u này s cho chúng ta và ch ng trình ch ti p c n c v i nh ng bi n khác c a l p ó.

Cho nh ng m c ích c a chúng ta, l p và c u trúc xây d ng ph i luôn luôn khai báo quy n s d ng **public** b t k b ph n c a d án c a b n có th t o ra m t th hi n c a l p (*create an instance of the class*)

Ngoài ra còn 1 vài ngoài l nh ng nó n m ngoài ph n t ng quan này.

Vehicle class

Quay tr l i vd, nh ng mā l nh s c thay i l i cho phù h p

```
1 //Vehicle.as
2 package {
3
4     import flash.display.MovieClip;
5     import flash.events.Event;
6
7     public class Vehicle extends MovieClip {
8
9         private var _gasMileage:Number;
10        private var _fuelAvailable:Number;
11        private var _milesTraveled:Number = 0;
12        private var _go:Boolean;
13        private var _tires:Tires;
```

Chúng ta quan tâm n dòng 1 nh 9 --> 13, i u này khai báo các bi n thu c lo i private (c hi u là nh ng thu c tính, ho c hàm, ho c ph ng pháp riêng cho m t l p nào ó, khōng ph i l p hay ch ng trình khác có th can thi p vào m t cách d dàng (chúng ta s th o lu n cách s d ng ngay bây gi) public hay private trong AS3 c hi u là namespaces trong c u trúc c a mā l nh)

ây là o n mā ti p

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER 'S GUIDE J

```
14
15     public function Vehicle(mpg:Number=21, fuel:Number=18.5) {
16         _gasMileage = mpg;
17         _fuelAvailable = fuel;
18         this.addEventListener(Event.ENTER_FRAME, onLoop, false,
19         0, true);
20     }
21
22     private function onLoop(evt:Event):void {
23         if (_go) {
24             _fuelAvailable--;
25             _milesTraveled += _gasMileage;
26             if (_fuelAvailable < 1) {
27                 this.removeEventListener(Event.ENTER_FRAME,
28                 onLoop);
29             }
30             trace(this, _milesTraveled, _fuelAvailable);
31             this.x = _milesTraveled;
32         }
33     }
34
35     public function go():void {
36         _go = true;
37     }
38
39     public function get gasMileage():Number {
40         return _gasMileage;
41     }
42
43     public function set gasMileage(mpg:Number):void {
44         _gasMileage = mpg;
45     }
46
47     public function get fuelAvailable():Number {
48         return _fuelAvailable;
49     }
50
51     public function set fuelAvailable(fuel:Number):void {
52         _fuelAvailable = fuel;
53     }
54
55     public function get milesTraveled():Number {
56         return _milesTraveled;
57     }
```

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER 'S GUIDE J

```
57     public function get tires():Tires {
58         return _tires;
59     }
60
61     public function set tires(Tires):void {
62         _tires = tires;
63     }
64 }
65 }
```

Mã 1 nh còn 1 i gi nguyên tr Ham onloop() à c t private (ho t ng riêng trong 1 p) v y làm sao ch ng trình có th s d ng c hàm onloop này t 1 p vehicle.as. Câu tr 1 i là s d ng hàm này b ng cách thông qua các hàm **get** và **set** c n ph i hi u **set** và **get** là 2 t khóa.

Cách 1 nh t dòng 37--> 63 à c thêm vào nh m xây d ng các hàm **get** và **set**, 2 hàm này b n có th ki m soát t t các thông tin s **ra** và **vào** trong nh ng thu c tính riêng (private) c a l p

Hàm go() c n gi nguyên là public chúng ta có th can thi p n nó t nhi u v trí t mã 1 nh khác trong ch ng trình.

Nh chúng ta à th o lu n, hàm get and set xác nh nói v nh ng ph ng pháp nh nhau

Hàm tr v giá tr hi n th i hay thi t l p giá tr m i cho thu c tính, s d ng cú pháp t ng t nh àn êu. Chúng c xem nh nh ng l i h ng d n v mă l nh, làm mă l nh tr lén d hi u, n u b n khôn g pha c p n get ho c set ho c th m chí s d ng cú pháp c a ph ng pháp ó b n v n có th làm mà khôn g c n óng gói tuy nhiên m t l i ích khác uyocj a ra sau ây

Chung ta t o ra nh ng KQ mong mu n v i l p, chúng ta có th s d ng nh ng thu c tính c ng nh ph ng pháp c a nó d dàng

```
vehicle.gasMileage = 10;
trace(vehicle.gasMileage);
```

Xem xét o n mă sau :

```
public function get gasMileage():Number {
    return _gasMileage;
}
```

C n nh giá tr tr v cho vi c s d ng thu c tính **_gasMileage** c a l p vehicle là **gasMileage**, t c là b n ch có th tham chi u n thu c tính này thông qua giá tr có tên là “[tên i t ng ang s d ng l p vehicle].gasMileage”

VD:

```
var compact:Car = new Car(21, 18);
trace(compact.gasMileage)
```

BIÊN SO N L IT SÁCH LEARNING ACTION SCRIPT BEGINNER 'S GUIDE J

21 c truy n cho tham s **_gasMileage** c a l p Car, l p này là l p m r ng c a Vehicle mà l p Vehicle l i có hàm **get gasMileage()**, do ó i t ng **compact** này s có th s d ng giá tr c a thu c tính **_gasMileage** thông qua bi n **compact.gasMileage**

Hàm get và set cho chúng ta truy c p, thay i các thu c tính c a l p, ó là m t cách t t. Tuy nhiên chúng ta c n m t cách truy su t tr c ti p h n n a,tr c ây chúng ta s d ng nh ng thu c tính thu c lo i private b ng cách xây d ng thêm m t subclass (b sung l p) dành cho vi c này. S d ng ph ng pháp super() là m t s 1 a ch n m i m nh ng r t hi u qu . Chúng ta nhìn th y ph ng pháp này dòng 6 c a 2 l p sau:

Car class

//Car.as

```
package {  
    import flash.display.MovieClip;  
    import flash.events.Event;  
    public class Car extends Vehicle {  
        public function Car(mpg:Number, fuel:Number) {  
            super(mpg, fuel);  
            var tires:Tires = new Tires("highperfomance");  
            trace(this + " has " + tires.type + " tires");  
        }  
        public function openSunroof() {  
            trace(this, "opened sunroof");  
        }  
    }  
}
```

Truck class

//Truck.as

```
package {  
    import flash.display.MovieClip;  
    import flash.events.Event;  
    public class Truck extends Vehicle {  
        public function Truck(mpg:Number, fuel:Number) {  
            super(mpg, fuel);  
        }  
    }  
}
```

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER 'S GUIDE

```
    var tires:Tires = new Tires("snow");
    trace(this + " has " + tires.type + " tires");
}
public function lowerTailgate() {
    trace(this, "lowered tailgate");
}
}
```

Làm super(mpg, fuel); truy cập 2 tham số **mpg** và **fuel** cho 2 biến **_gasMileage** và **_fuelAvailable** 1 lớp vehicle, vì 2 thuộc tính này là private nên chỉ 1 lớp vehicle có quyền can thiệp, sau đó nó truy cập thuộc tính này cho cả chương trình chỉ ng trình hiện đang 2 mặt lớp **compact** và **pickup** có 2 biến **_gasMileage** và **_fuelAvailable** mang giá trị c truy cập. Vì cách làm này là an toàn cho dữ liệu 1 lớp vehicle – chúng ta nói rõ ràng về phép này trong phần sau.

Mã lệnh hàm **Tires.as** và main flash vẫn như cũ.

Polymorphism

(tính đa hình)

Đây là khái niệm mà chúng ta bàn luận trong lập trình OOP, là khả năng chỉ có khía cạnh hay không chỉ có khía cạnh (subclass) bao gồm hàm, phương pháp của lớp chính (superclass) đang cho nó khả năng, hoặc chỉ có khía cạnh, phương pháp nào khi chúng có tên trùng nhau... (VD trong 2 lớp, một lớp thừa kế từ lớp kia nhưng 2 lớp đều có 2 phương pháp có tên giống nhau nhưng chức năng khác nhau, tính đa hình sẽ cho chúng ta biết cách sử dụng 2 hàm này tùy theo ý của chúng ta mà flash không bị nhầm lẫn)

Mã lệnh của chúng ta không có gì thay đổi, đây là lớp **Vehicle.as**

Vehicle class

//Vehicle.as

package {

```
import flash.display.MovieClip;
import flash.events.Event;
public class Vehicle extends MovieClip {
    private var _gasMileage:Number;
    private var _fuelAvailable:Number;
    private var _milesTraveled:Number = 0;
    private var _go:Boolean;
```

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER 'S GUIDE

```
private var _tires:Tires;
public function Vehicle(mpg:Number=21, fuel:Number=18.5) {
    _gasMileage = mpg;
    _fuelAvailable = fuel;
    this.addEventListener(Event.ENTER_FRAME, onLoop, false,
        0, true);
}
private function onLoop(evt:Event):void {
    if (_go) {
        _fuelAvailable--;
        _milesTraveled += _gasMileage;
        if (_fuelAvailable < 1) {
            this.removeEventListener(Event.ENTER_FRAME,
                onLoop);
        }
        trace(this, _milesTraveled, _fuelAvailable);
        this.x = _milesTraveled;
    }
}
public function changeGear():void {
    trace(this, "changed gear");
}

public function useAccessory():void {
    trace(this, "vehicle lights turned on");
}
public function go():void {
    _go = true;
}
public function get gasMileage():Number {
    return _gasMileage;
}
public function set gasMileage(mpg:Number):void {
    _gasMileage = mpg;
}
public function get fuelAvailable():Number {
```

```
        return _fuelAvailable;
    }
public function set fuelAvailable(fuel:Number):void {
    _fuelAvailable = fuel;
}
public function get milesTraveled():Number {
    return _milesTraveled;
}
public function get tires():Tires {
    return _tires;
}
public function set tires(tires:Tires):void {
    _tires = tires;
}
}
```

Mã 1 nh thêm vào đây là

```
public function changeGear():void {
    trace(this, "changed gear");
}
```

```
public function useAccessory():void {
    trace(this, "vehicle lights turned on");
}
```

ây không ph i là 2 hàm xa l , trái l i chúng ta ã s d ng nó nhi u, bây gi chúng ta dùng nó nh m t ví d minh h a cho tính a hi u Polymorphism, hai l p Car và Truck c ng có th s d ng 2 hàm nàu theo tính th a k c a OOP
Tính a hi u này cho phep chúng ta s d ng các hàm, ph ng pháp c a m t l p th a k cho 1 p chúng ta ang s d ng. th a k t superclass

Chúng ta xem xét 1 p Car và Truck m i:

//Car.as

```
package {
    import flash.display.MovieClip;
    import flash.events.Event;
    public class Car extends Vehicle {
        public function Car(mpg:Number, fuel:Number) {
```

BIÊN SO N L I T SÁCH LEARNING ACTION SCRIPT BEGINNER 'S GUIDE

```
super(mpg, fuel);
var tires:Tires = new Tires("highperfomance");
trace(this + " has " + tires.type + " tires");
}
public function openSunroof() {
    trace(this, "opened sunroof");
}
override public function useAccessory():void {
    openSunroof();
}
}
```

M i th v n nh c tr l p Car à thêm hàm **useAccessory()** i u náy bình th ng thì không có v n jì nh ng trong tr ng h p na , l p vehicle c ng có m t hàm có tên t ng t , v y tránh nh m l n, chúng ta s d ng t khóa **override** cho hàm **useAccessory()** , t khóa ó có ngh a là “ è” hàm **useAccessory()** s n có trong l p Vehicle mà thay vào ó s th c hi n hàm **useAccessory()** c a l p Car – 100% mă l nh c a l p vehicle à b è, tuy nhiên m t s tr ng h p chúng ta v n mu n gi l i mă l nh t hàm **useAccessory()** c a vehicle, chúng ta xem xét VD c a Truck.

```
//Truck.as
package {
    import flash.display.MovieClip;
    import flash.events.Event;
    public class Truck extends Vehicle {
        public function Truck(mpg:Number, fuel:Number) {
            super(mpg, fuel);
            var tires:Tires = new Tires("snow");
            trace(this + " has " + tires.type + " tires");
        }
        public function lowerTailgate() {
            trace(this, "lowered tailgate");
        }
        override public function useAccessory():void {
            lowerTailgate();
            super.useAccessory();
        }
    }
}
```

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE

```
}
```

L nh **super.useAccessory();** s th c hi n hàm **useAccessory();** t Vehicle.as t c là t hàm superclass c a nó.

Mã l nh file main flash:

```
var compact:Car = new Car(21, 18);
compact.x = 10;
compact.y = 20;
addChild(compact);
compact.changeGear();
compact.useAccessory();
```

```
var pickup:Truck = new Truck(16, 23);
pickup.x = 10;
pickup.y = 100;
addChild(pickup);
pickup.changeGear();
pickup.useAccessory();
```

```
stage.addEventListener(MouseEvent.CLICK, onClick, false, 0, true);
```

```
function onClick(evt:MouseEvent):void {
    compact.go();
    pickup.go();
}
```

Nh n Ctrl + Enter so sách s khác bi t

```
[object Car] has economical bias-ply tires
[object Car] changed gear
[object Car] opened sunroof
[object Truck] has storm-ready snow tires
[object Truck] changed gear
[object Truck] lowered tailgate
[object Truck] turned on lights
[object Car] 21 17
[object Truck] 16 22
[object Car] 42 16
[object Truck] 32 21
...
```

Compact không có dòng “turned on lights” vì hàm ghi ra nó à b “ è”

Navigation Bar Revisited

(làm 1 i thanh t ng tác)

* menu nút l nh

ch ng 4 chúng ta ā có m t s ánt ng t , nh ng chúng ta m i ch dùng mã l nh, bây gi chúng ta s v n d ng nh ng c tính c a OOP xâyd ng m t thanh t ng tác m i.

B n hây m h s LAS3Lab.fla bi t d ánc a chúng ta s nh th nào.

Trong d ánc a chúng ta, MovieClip trông gi ng nh m t thanh tab, m i nút th c n s n m trên thanh t ng tác. Tên c a m i nút l nh chúng ta có th i u khi n b ng MC_label nó c l y thông tin t l p có v trí **app.gui.MenuButtonMain**.

* ng vi n.

Có chi u rông 8 pixel và chi u dài tùy vào chi u dài c a nút l nh, nó s bao l y nui l nh. Nó c tac ra m t cách ng (t c là dùng mã l nh) l p có v trí **app.gui.HlineThick**. N u b n mu n t làm 1 d ánm i, t t nh t hây theo a ch c a các l p trong file m u, và s d ng tài nguyên c a b n theo ý mu n. b n c ng có th phát tri n file m u theo ý c a b n.

DOCUMENT CLASS

M c tiêu u tiên c a chúng ta là xâyd ng l p LAS3Main.as Hây nó liên k t ch t ch v i file h s fla b ng cách chúng tôi ā ch cho các b n chapter 1. Chúng ta nh p vào l p LAS3Main.as theo l p c a **flash.display.Sprite** và nó th a h ng c tính c a sprite

//LAS3Main.as

package {

 import flash.display.Sprite;

 import app.gui.NavigationBar;

//nh p mā ngu n c a l p NavigationBar

 public class LAS3Main extends Sprite {

 public function LAS3Main() {

 var appData:Array = ["one", "two", "three", "four", "five"];

//t o m ng có 5 ph n t ki u chu i ký t

 var navBar:NavigationBar = new NavigationBar(this, appData);

// t o m t i t ng theo l p NavigationBar (l p này s c tìm hi u sau)

 addChild(navBar);

//Thêm nó vào danh sach màn hình (display list – nghe quen không ? K)

}

}

Sprite c ng khôn khíc nhì u so v i MovieClip, d i góc c a phát tri n game thì sprite là m t v t th có th di chuy n và t ng tác trong game. Th c t v i AS2 ho c th m chí là AS3 tr xu ng, ng i ta v n s d ng MovieClip làm 1 sprite khi vi t game flash.

L p NavigationBar

Chúng ta s vi t ti p l p này, à c c p ph n trên :

// app > gui > NavigationBar.as

package app.gui {

import flash.display.Sprite;

public class NavigationBar extends Sprite {

//l p này k th a thu c tính c a sprite

private var _app:Sprite;

private var _hline:HLineThick;

//l p HlineThick s c vi t sau

private var _navData:Array;

public function NavigationBar(app:Sprite, navData:Array) {

//hàm này s c nh n tham s truy n vào t l p LAS3Main.as

// theo câu l nh: var navBar:NavigationBar = new NavigationBar(this, appData);

_app = app;

_navData = navData;

build();

//hàm build(); này s c vi t ngay sau ây

}

private function build():void {

for (var i:uint; i < _navData.length; i++) {

//t o s nút b m = giá tr thu c tính _navData.length = 5 ph n t

var menuBtn:MenuButtonMain = new MenuButtonMain(navData[i]);

//l p MenuButtonMain s c nh ngh a sau

menuBtn.x = 20 + (menuBtn.width + 2) * i;

menuBtn.y = 75;

//t a c a nút b m cách l trái 20 pixel, m i nút k nhau cách nhau 2 pixel

addChild(menuBtn);

//thêm vào màn hình

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER 'S GUIDE

```
        }  
//kết thúc vòng lặp for  
    _hline = new HLineThick();  
//tóm tắt: ngang mìn lú ý là thanh ngang chúng ta sẽ tạo theo  
màu //it's not có trong library có các thuộc tính đặc biệt (như tệp linkage  
là //HLineThick)  
    _hline.y = 100;  
    _hline.mouseEnabled = false;  
//thanh ngang dài các nút trên thanh toolbar  
//Chuột bô vô hiệu hóa khi rê chuột và click vào thanh toolbar mouse  
feedback // (phản hồi) and event trapping (bắt) – khi click vào thanh ngang  
chắc có ý nghĩa //gì  
    addChild(_hline);  
//thêm vào màn hình  
}  
//kết thúc hàm build  
}  
//kết thúc lớp NavigationBar  
}  
//kết thúc gói
```

MenuButtonMain

(lập MenuButtonMain)
Lập MenuButtonMain như sau:

```
package app.gui {  
    //tạo một lớp  
    import flash.display.Sprite;  
    import flash.text.TextField;  
    import flash.events.MouseEvent;  
    //nêu các class cần thiết có sẵn trong flash và text, event ...  
    public class MenuButtonMain extends Sprite {  
        public var _label:TextField;  
        //TextField là một kiểu dữ liệu có sẵn trong flash  
        public function MenuButtonMain(label:String) {  
            //chúng ta xem lại var menuBtn:MenuButtonMain = new MenuButtonMain(navData[i]);
```

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE

```
//t 1 p NavigationBar, v i vòng l p for ó + L p MenuButtonMain này, chúng ta
s //t o ra c 5 nút có label l n l t là “one”, “two”,... “five” và nó g n cho
_label.text
    _label.text = labl;
// n i d ng c a l p text chính là n i dung c a tham s labl truy n vào.
    _label.mouseEnabled = false;
//t t các s ki n c u chu t khi rê trên _label
    buttonMode = true;
//có th s d ng nh m t button (ch kh ng ch là MovieClip)
    useHandCursor = true;
//cho phép s d ng chu t hình bàn tay khi rê vào Nút b m
    addEventListener(MouseEvent.CLICK, onClick, false, 0, true);
}
//K t thúc hàm MenuButtonMain
private function onClick(evt:MouseEvent):void {
    trace(_label.text);
//N i dung c a _label.text hi n lên khi ta click vào nó
}
//k t thúc hàm onClick
}
// k t thúc l p MenuButtonMain
}
//k t thúc gói
```

B SUNG CHAPTER 6

(VD v Navigation Bar c a ch ng này khá khó hiểu, các b n có th xem VD sau c a mình, t t nhiên là n gi n h n – minh t vi t mà – hi v ng các b n s hi u h n v oop)

D án c a chúng ta s là xây d ng các sprite trên màn hình có kh n ng t ng tác b ng cách vi t l p cho nó.

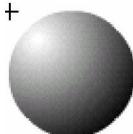
u tiên chúng ta t o m t movieclip:

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER 'S GUIDE J



Vì hình tròn, sau đó tô màu cho nó giống quả bóng. Vì thế này cho nó 3D một chút

J
+



Chúng ta viết thêm lớp cho MovieClip này: (lưu dưới tên Ball.as)

```
package {  
//cau truc gói  
  
import flash.display.MovieClip;  
import flash.events.Event;  
import flash.events.MouseEvent;  
public class Ball extends MovieClip {  
    public var dx:Number = Math.random()*10 - 5;  
    public var dy:Number = Math.random()*10 - 5;  
//khai báo các giá trị dx, dy là tốc độ chuyển của quả bóng.  
    public function Ball() {  
        x = Math.random()*500;  
        y = Math.random()*400;  
//    tọa trí ngẫu nhiên của quả bóng trên màn hình  
        addEventListener(MouseEvent.CLICK, onClick, false, 0, true);  
}
```

BẢN ĐỒ CHI TIẾT I HUYETSATHIENHA – thanh_vinh648@yahoo.com
12/1 – THPT QUỐC HỘI - BẢN ĐỒ CHỈ LƯỜNG HÀNH TRÊN <http://WWW.VNFX.COM>

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER 'S GUIDE J

```
        addEventListener(Event.ENTER_FRAME, onLoop, false, 0, true);
//thêm 2 s kiên c a FRAME_FRAME và MOUSE     th c hi n 2 hàm onClick() và onLoop()
    }
// nh ngh a hàm onLoop
    private function onLoop(evt:Event):void {

        x += dx;
        y += dy;
//Qu bóng d ch chuy n theo 2 giá tr dx và dy
        if ((x > 550-width) || (x < 0)) {
            dx *= -1;
        }
        if ((y > 400-height) || (y < 0)) {
            dy *= -1;
        }
//khi qu bóng ch m n n thì nó s b t ng c l i do giá tr dx (ho c dy) b o ng c
// chung ta vi t 550-width t c là giá tr r ng c a màn hình - chi u dài qu bóng (giúp
qu //bóng ph n h i gi ng th t h n
        }
// nh nghĩa hàm onClick() xóa qu bóng khi chúng ta click vào nó
    private function onClick(evt:Event):void {
        removeChildAt(0);
    }
}
//k t thúc goi'
```

ây là mã l nh t o ra 100 qu bóng, vi t trong Main Flash

```
for (var inc:uint = 0; inc < 100; inc++) {
```

```
    var ball:Ball = new Ball();
```

```
    addChild(ball);
```

```
}
```

K T LU N:

Vì sao mình l i a ra VD này ?, là vì chúng ta có th th y c 1 ng d ng r t hay
c a l p là chúng ta có th t o ra nhi u b n sao c a m t nguyên b n nào ó mà có m t
l p mô t chi ti t cho nó nh ng không có file m u nào trong sách nói n i u này.

i u này th c s có ý ngh a (VD khi b n vi t 1 game mu n t o 100 con quái thì m
l p này r t h u ích v i b n, ng th i chúng ta có th thay i, m r ng l p có
thêm nhi u thu c tính c ng nh ch c n ng m i c a quái ch ng h n (ã c nói
trong chapter 6)



(S chuy n ng)

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE J

T kinh nghi m u tiên n khi m tr t nhi u th i gian, bây gi b n ã có th d ch chuy n m t i t ng nào ó l y t tài nguyên c a b n (bitmap ch ng h n --> movieClip...) b ng mă l nh. B n có th ã hài lòng v i u ó. B n có th so sánh vi c s d ng AS v i cách làm các i t ng chuy n ng theo các công c s n có c a flash,s d ng AS ch ng nh ng nó giúp b n gi i phóng kh i timeline dài l ng nh ng mà kém hi u qu khi xây d ng chuy n ng theo các công c ho t hình c mà còn cho phép b n i u khi n i t ng m t cách hoàn ch nh, có th t do ho t d ng theo ý mu n. Chúng ta s d ng AS mô ph ng m t s di chuy n s có th làm cho nó gi ng th t h n thông qua vi c t ng tác v i các mô tr ng có s bi n i a dang. Các b n có th s d ng h s m u theo dõi ch ng này và phát tri n nó theo nh ng d án riêng c a b n. B n c n ph i có ki n th c nh t nh v toán h c, bao g m l ng giác (h i khó v i HS – mình c ng là HS mà), CT Py-ta-go (cái này d c),... tuy nhiên n u b n hi u rõ v ng d ng c a toán h c trong l p trình chuy n ng thì có ngh a là b n ã i c m t b c dài.

M bài h i dài – song i v i sách thì nó còn dài h n J.

Basic Movement

(s di chuy n n gi n)

n gi n n m c các b n c ng ã bi t, chúng ta s s d ng mă l nh m t i t ng di chuy n thông qua thu c tính **t a x và y** c a nó. Flash có m t h tr c t a v i tâm t a là nh c a vùng làm vi c phía trên bên trái c a vùng. Tr c Ox h ng ngang chi u d ng t trái--> ph i, tr c Oy h ng th ng ng, chi u d ng h ng xu ng. Cái này h i khác trong toán h c m t tí :D.

Gi s chúng ta có 2 movieclip là **mc** và **mc2**, chúng ta cho chúng di chuy n b ng cách:

mc.x++;

mc.y--;

mc2.x += 10;

mc2.y -= 10;

Câu h i là nó s di chuy n c bao nhiêu? **mc** s qua ph i 1 pixel và lên trên 1 pixel --> nó s i sang h ng ông b c 1 góc 45^0 và quãng ng i c là c n b c 2 c a 2. **mc2** s qua ph i 10 pixel và lên trên 10 pixel --> nó s i sang h ng ông b c 1 góc 45^0 và quãng ng i c là c n b c 2 c a 200 (theo nh lý py-ta-go). V n là khi nó di chuy n xong thì s nh th nào, cái này tùy vào mă l nh ang c t âu, n u trong timeline chính thì sau khi di chuy n xong nó s d ng l i. N u t

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE J
trong m t hàm nào ó có EventListener là ENTER_FRAME thì nó sẽ di chuyển mãi mãi.

VD:

```
var ball:MovieClip = new Ball();
ball.x = ball.y = 100;
addChild(ball);
var xVel:Number = 4;
var yVel:Number = 4;
addEventListener(Event.ENTER_FRAME, onLoop, false, 0, true);
function onLoop(evt:Event):void {
    ball.x += xVel;
    ball.y += yVel;
}
```

Khi quả banh đi quá màn hình, bỗng nhiên nó bị ném không phải vì nó không chuyển sang hướng mà nó đang chuyển sang vùng mà chúng ta không thấy được (và sẽ không thấy được nữa).

Bên cạnh đó ta có thể xác định vận tốc (x,y) cái này nói rồi, ta có thể tính bằng ($quãng\ đường\ đi\ c/c$)/(thời gian di chuyển ò) th c t
vận tốc trong các đơn vị m/s

VD: Cứ sau 1 giây thì mỗi 1 nh sau sẽ di chuyển khi r i vào s khi n ENTER_FRAME:

```
ball.x += 10;
```

vận tốc đây là 10pixel/giây. Nếu ta thấy ng coi vận tốc là một vector có hướng và giá trị là 1 m/s. Vectors vận tốc chính là 10pixel/giây còn hướng của nó là 1 m/s. Khi ta chỉ sử dụng trục Ox (hoặc Oy) thì ta có giá trị là 1 m/s. Ta nói vận tốc là một vector có hướng và giá trị là 10pixel/giây.

Chúng ta xem xét n 1 khai niêm vận lý nào, ó là giá trị, mà con tàu v bay vào không gian thì không có vận tốc là 1 m/s, chỉ có vận tốc là 0 m/s. Thời gian là 1 giây là 10pixel/giây (g = 9.8 m/s²), vậy giá trị là cái gì và vì sao nó có vận tốc là 0 m/s². Ta thấy là 1 hàm bao gồm thời gian là 1 giây và là 1 hàm bao gồm thời gian là 10pixel/giây.

VD:

```
var ball:MovieClip = new Ball();
ball.x = ball.y = 100;
addChild(ball);
var xVel:Number = 4;
```

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER 'S GUIDE J

```
var yVel:Number = 4;  
var xAcc:Number = 1;  
var yAcc:Number = 1;  
//khai báo 2 giá trị gia tốc theo trục Ox và Oy  
addEventListener(Event.ENTER_FRAME, onLoop, false, 0, true);  
function onLoop(evt:Event):void {  
    ball.x += xVel;  
    ball.y += yVel;  
    xVel += xAcc;  
    yVel += yAcc;  
    //Gia tốc sẽ tăng vitesse n tại c lên (+ thêm xVel và yVel) mỗi khi câu lệnh c  
    //được thực hiện // ở vitesse n tại c tăng theo m thời gian b có H số góc là xAcc (hoặc là  
    //yAcc). Các biến này có sẵn toán tử 9 có nói về Hs số góc :D  
}
```

Mã lệnh không khác gì $v = v_0 + at$ mà chỉ thêm vào. Công thức tính gia tốc ($v - v_0$ sau - v_0 trước)/(thời gian t vì có a t ng).

$V = v_0 + at \rightarrow$ gia tốc có $v = (m/s)/s = m/s^2$.

VD: sau một khoảng 5s vận tốc tăng từ $v = 10\text{ m/s}$ lên 20 m/s thì gia tốc là bao nhiêu?

$a = (v_f - v_i)/t = (20 - 10)/5 = 2\text{ m/s}^2$ tức là 2 m/s^2 trong khoảng thời gian 5s.

Gia tốc v là ta có thể hiểu là $\frac{\Delta v}{\Delta t}$ theo một vect có hướng và lối đi.

VD mã trên gia tốc chính là **xVel** và **yVel**. Bởi nó có thể sử dụng quy tắc Pythagoras để tính vect gia tốc chính xác (có $\sqrt{v_x^2 + v_y^2}$, hoặc là $\sqrt{xVel^2 + yVel^2}$).

Geometry and Trigonometry

(hình học và lượng giác học)

Distance

(khoảng cách)

Đầu tiên, hãy nói về việc giải bài toán lập trình 1 game, trong game sẽ có kí tự truy cập và bắn phím tìm cách thoát. Khi kí tự quá gần, bạn bước vào cách nhau mảng thông tin thoát hi vọng không trong 2 mảng. Nếu cách kí tự khi nhận vào và bắn phím thoát kí tự có thể bắt cung cấp cho anh ta

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER 'S GUIDE J
 có m t tính toán sai l m, t o c i u ó, k thù c a chúng ta c ng ph i cho k thù bi t c n i thoát hi m g n nh t c a ng i ch i.

Let's say you are programming a game in which a character must be pursued by an enemy and must exit through one of two doors to safety. However, the enemy is close enough that the character must choose the nearest exit to survive. The player controls the character, but you must make the game challenging enough for the enemy to catch the character if the player makes the wrong decision. To do that, the enemy must know which exit is closest.

V n c a chúng ta là làm sao k thù c a ng i ch i bi t trong 2 l i thoát, l i nào g n h n? – và nh lý Py-ta-go s cho chúng ta câu tr l i. nh lý này nói r ng: chi u dài c a c nh dài nh t c a m t tam giác vuông b ng c n b c 2 c a t ng bình ph ng 2 c nh còn l i

Theo chi u ngang và chi u d c (the theorem says that the length of the longest side of a right triangle is equal to the square root of the sum of the squares of the horizontal and vertical sides)

Rightangle: góc vuông

Hypotenuse: c nh huy n

Hình 7-3 cho chúng ta cách tính kho ng cách gi a 2 i m b t k theo t a c a chúng b ng nh lý Py-ta-go.

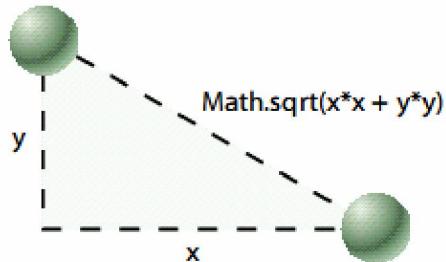
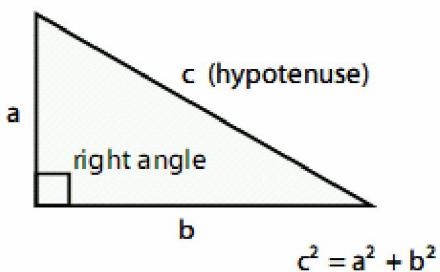


Figure 7-3. Calculating the distance between two points using geometry

BIÊN SO N L I T SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE

Sau đây là một cách trình thuật và cách tính khoảng cách này, các bạn nên xem file mẫu để chúng tôi hoàn thành dự án như thế nào, trong đó có toàn bộ tài nguyên cần thiết mà vì có thể chúng nằm ngoài phạm vi nội dung của cuốn sách: Hàm tính khoảng cách:

```
function getDistance(x1:Number, y1:Number, x2:Number, y2:Number):  
Number {  
    var dx:Number = x1-x2;  
    var dy:Number = y1-y2;  
    return Math.sqrt(dx * dx + dy * dy);  
}
```

Đây là một ví dụ cách sử dụng hàm getDistance() kèm theo file nguồn, nó là 1 số so sánh khoảng cách giữa ball1 và ball0 và giữa ball2 và ball0

Here is an example usage of our getDistance() function, seen in the accompanying distance1.fla source file. It compares the distance between ball0 and ball1 to the distance between ball0 and ball2:

```
var dist1 = getDistance(ball0.x, ball0.y, ball1.x, ball1.y);  
var dist2 = getDistance(ball0.x, ball0.y, ball2.x, ball2.y);  
if (dist1 < dist2) {  
    trace("ball1 is closest to ball0");  
} else {  
    trace("ball2 is closest to ball0");  
}
```

Movement Along an Angle

(Chuyển động theo một góc)

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER 'S GUIDE J

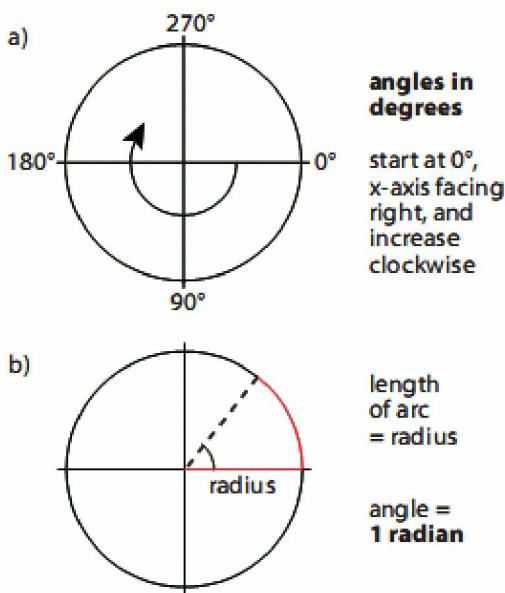


Figure 7-4. How Flash angles (a) and radians (b) are calculated

Phân này mình khá rành :D, vì ang ôn thi nên m y cái l ng giacute và goc này d nh tr bàn tay :D (hi v ng khong vi d ch sach mà r t H :D). Do ó mình v a d ch v a t ph bi n ki n th c mà mình bi t các b n d hi u, ng nói là mình o sach nghen K.

Chúng ta th y trên hình 7-4 là cách góc trong flash c a m t i t ng b t k , VD: b n có 1 movieclip **mc** có góc là $0^\circ \Rightarrow$ nó s h ng v phia bên ph i theo chi u tr c Ox, n u **mc** có góc là 270° Nó s h ng lên trên, ng c chi u c a tr c Oy.

Thu c tính góc c a i t ng chúng ta có th s d ng thu c tính **rotation** thay i m t cách d dàng.

ieu th 2 chúng ta bàn lu n là n v s d ng góc. Ph bi n nh t là và **radian**. Flash s d ng n v là radian trong tính toán (dùng trong hàm sin, cos, tan, atan ...) và chúng ta c n ph i bi t chuy n i qua l i gi a 2 n v này m t cách thành th c ti n cho công vi c l p trình.

$$1 \text{ Radian} = 180^\circ$$

M c dù i u này úng trên th c nghi m nh ng nguyên t c thì l i khong n v d u ng thuc x y ra v i 2 ai l ng có n v khác nhau

Ó Tuy nhiên chúng ta có th s d ng CT sau 1(Radian) = $180(\text{ n v là })/\pi$, g n úng là 57° . Trong AS chúng ta vi t nh sau:

Var degrees:Number = 180;

Var radian:Number;

radian = (degrees*Math.PI)/180;

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER 'S GUIDE J

//radian =

T degrees (n v là), chúng ta t o ra radian (n v là radian).

T ng t chuy n m t n v t radian sang , chúng ta có CT sau:

$$\text{Đ} 1 () = 1 \text{ (radian)} / 180;$$

Var degrees:Number ;

Var radian:Number= Math.PI;

degrees = radian * 180 / Math.PI;

//degrees = 180

2 VD trên là minh t thêm vô cho các b n d dàng khi l p trình, có trong sách nh ng nó không rõ ràng l m J .

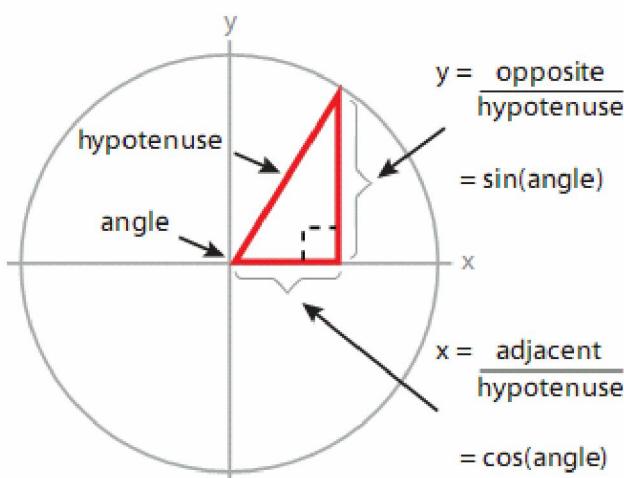


Figure 7-5. Four angles rotating around a circle, expressed both in degrees and x and y points on a circle with a radius of 150 pixels

Hình mà các b n th y trên là vòng tròn 1 ng giá bi u di n các giá tr c a sin và cos trong 1 ng giác.

Cho 1 tam giác vuông b t k thì:

$\sin(\text{góc nào ó}) = \text{C nh i c a góc}/\text{C nh huy n}$

$\cos(\text{góc nào ó}) = \text{C nh k c a góc}/\text{C nh huy n}$

Mình chú ý m t tí là ng tròn 1 ng giác c a hs ph thông thì c nh huy n chính là **hypotenuse** = 1 do ó y = sin(angle) hay tr c y th ng c g i là tr c sin, t ng t tr c Ox c g i là tr c cos). Chúng ta.

Chúng ta bàn lu n m t tí v vect cái ā, cái này trong sách không nói âu. ph n tr c mình có nói qua roài. Vect là m t i l ng có h ng, nó có l n nh t nh (là b ng chi u dài c a vect khi bi u di n trên m t ph ng b ng m t m i tên). M t vect luôn ch rō i u u (xu t phát) và i m cu i(m i c a m i tên)

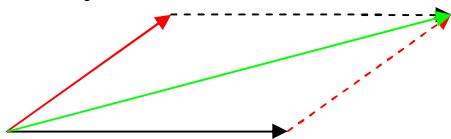
VD: $\frac{(1 \text{ n a})}{\rightarrow}$

(u) (cu i)
 (l n 2a có chí u dài g p ôi cái trên)

(u) (cu i)

Chúng ta có th t ng h p 2 vect b ng nhi u quy t c và các tr ng h p khác nhau

1. Quy t c Hình bình hành:



Nh ng ng g ch t khôn ph i là vect mà là minh h a cho cái hình bình hành thình. :D

t ng h p 2 vect b t k (màu và en) và vecto t ng h p là màu xanh, chúng ta th y r ng vecto t ng h p chính là hình chéo c a hình bình hành có 2 c nh có l n là b ng l n c a 2 vect c n t ng h p.

Công th c này dành cho HS l p 10 v t lý thoai, mình a ra công th c sau tính dài

C a vecto xanh.

$$\text{1 n vect : Xanh}^2 = \text{en}^2 + \text{en}^2 - 2 * \text{en} * \text{en} * \text{Cos}$$

(= góc gi a vect khôn t o n và en t o n)

L y c n b c 2 chúng ta có 1 n vect xanh t ng h p.

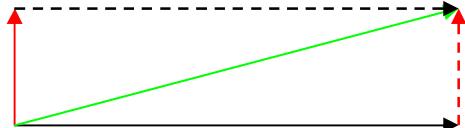
Các b n có th tính h ng c a vect t ng h p

2. Công th c tính 2 vect vuông góc v i nhau

Th c t góc gi a 2 vect trong flash theo Ox và Oy là 90° chúng ta t ng h p vect theo Công th c sau:

$$\text{Xanh}^2 = \text{en}^2 + \text{en}^2$$

(tr ng h p này = $90^\circ \Rightarrow \cos = 0 \Rightarrow 2 * \text{en} * \text{en} * \text{Cos} = 0$)



Vì sao ph i h c cách t ng h p vect ? – là vì t c (ho c t ng t các b n có th bi u di n gia t c) trong flash th c ra **chính là t ng h p c a 2 vect v n t c c a 2 tr c Ox và Oy**, n u b n chuy n ng qua ph i 2 pixel và xu ng d i 2 pixel thì chính là b n ang di chuy n theo h ng ông nam và v i v n t c c n b c 2 c at ng ($2^2 + 2^2$)

minh h a, chúng ta s nghiên c u VD sau: cho m t v t có m t v n t c nh t nh, hãy bi u di n chuy n ng c a nó trên màn hình.

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE

u tiên, v n t c là m t i l ng vect , cho v n t c t c là cho chúng ta h ng (góc c a nó) và dài – l n (là t c c a v n t c).

bì u di n chuy n ng trong flash, chúng ta c n bi t các giá tr **xVel** và **yVel**, v t có th chuy n ng c (trong ph n tr c có nói roài). Chúng ta s d ng mă l nh sau:

```
var ball:MovieClip = new Ball();
ball.x = ball.y = 100;
addChild(ball);
var speed:Number = 12;
var angle:Number = 45;
//các c tính c b n c a vect v n t c
var radians:Number = deg2rad(angle);
//chuy n i t angle ( n v là ) sang radian
var xVel:Number = Math.cos(radians) * speed;
//xVel ph i nhân cho speed bi t dài c a vect vân t c theo tr c Ox, n u khôn g nhân //thì chúng ta ch có v n t c c a vect v n t c c b n (c nh huy n c a tam giác trong // ng tròn 1 ng giác = 1 n v ) có dài là 1*radians
var yVel:Number = Math.sin(radians) * speed;
//yVel ph i nhân cho speed bi t dài c a vect vân t c theo tr c Oy, n u khôn g nhân //thì chúng ta ch có v n t c c a vect v n t c c b n (c nh huy n c a tam giác trong // ng tròn 1 ng giác = 1 n v ) có dài là 1*radians

//các giá tr vect theo 2 tr c Ox và Oy (t ng h p l i chính là vect v n t c) v t di
//chuy n theo 2 tr c t a
addEventListener(Event.ENTER_FRAME, onLoop, false, 0, true);
function onLoop(evt:Event):void {
    ball.x += xVel;
    ball.y += yVel;
    //ball di chuy n.
}
function deg2rad(deg:Number):Number {
    return deg * (Math.PI/180);
    //tr v giá tr c a radian
}
```

Circular Movement

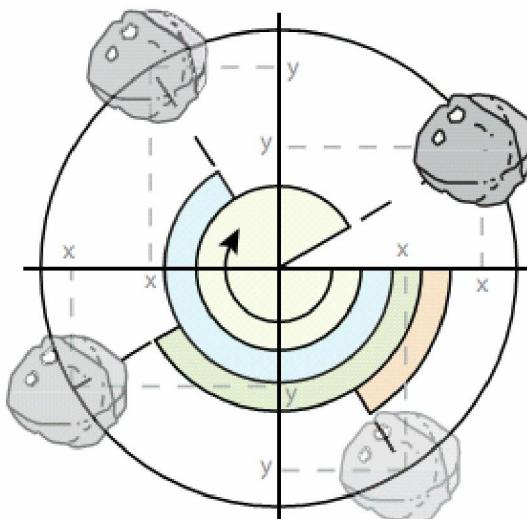
(s di chuy n theo ng tròn)

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER 'S GUIDE ↗

Phân này khá khó, sử dụng giác auуль m. :))

Cái hình này b vào cho sinh ng thoại, ch nó khó hiểuуль m

- 50°
x: `Math.cos(50) = 0.96 (x 150 = 144)`
y: `Math.sin(50) = -0.26 (x 150 = -39)`
- 140°
x: `Math.cos(140) = -0.20 (x 150 = -30)`
y: `Math.sin(140) = 0.98 (x 150 = 147)`
- 230°
x: `Math.cos(230) = -0.79 (x 150 = -118.5)`
y: `Math.sin(230) = -0.62 (x 150 = -93)`
- 320°
x: `Math.cos(320) = 0.90 (x 150 = 135)`
y: `Math.sin(320) = -0.43 (x 150 = -64.5)`



Bên chuyền quay nhau ng góc khác nhau xung quanh một vòng tròn, biến thành hai mảnh trong nhau “ ” c a góc. Và hai là tia X và y chia trên và mảnh tròn v i mảnh bán kính 150 pixel.

Bên ảnh biến cách xác nh trên trục x và y thông qua giá trị của vector vận tốc (Speed, hướng (angle)), chúng ta có thể phát triển lên chuyền quay tròn.

Lấy VD n gianh trái t chuyền quay tròn quanh mảnh trái, và mảnh tròn chuyền quay tròn quanh trái t (tròn đây có tính tảng i, thết thì nó chuyền quay hình elip :D)

Chúng ta không quan tâm đến lối cách vận tốc hay hướng của nó vì chúng ta không sử dụng vector biến đổi chuyền quay, mà vector chỉ biến đổi 1 hướng trong khi chuyền quay tròn thì hướng chuyền quay ngang cava và là biến thiên., thay vào đó chúng ta sẽ tính toán các giá trị x và y liên tiếp trong những góc khác nhau, bằng cách sử dụng hàm sin và cos, bạn có thể di chuyển mảnh theo hình tròn.

Kết thúc chúng tôi trình bày không quá khó hiểu, nhưng vẫn dùng ứng các hàm sin và cos trong những góc khác nhau, giá trị của sin hay cos thu được [-1;1]. Vòng tròn lồng giác có bán kính là 1. tì (0;1), chúng ta có sin = 0 và cos = 1, mô típ mìn ngoài cùng bên phải, tảng tiv i các mảnh khác nhau.

Câu lệnh sau chính là trái tim của chương trình:

```
satellite.x = centerX + radius * Math.cos(radian);
satellite.y = centerY + radius * Math.sin(radian);
```

các lệnh này để giác cho chúng ta thấy rõ ràng, **bán kính** **ng tròn chuyền quay** ***cos(góc cava tò)** chỉ u lên trục cos trong ng tròn lồng giác chính là tia x (mình đã nói trục Ox gòn gọi là trục cos ứng không nào ?) + tia c a

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE

trung tâm là m t h ng s , s d c n công thêm vì n u không c ng, v t ch chuy n ng xung quanh ng tròn l ng giác c b n có tâm là g c t a (0;0) mà thôi, chính xác h n thì ây là phép t nh ti n g c t a theo vect (centerX:centerY), các b n thông c m, c hi u v y thôi, n u bây gi mìn trình bày thêm v phép tính ti n vect thì v a m t th i gian mà không úng tr ng tâm bài h c. gi i thích t ng t cho dòng l nh th 2.

Mã l nh m t v t chuy n ng là ây, các b n tham kh o file m u bi t d án c xây d ng nh th nào

```
var angle:Number = 0;  
//góc ban u c a v t  
var radius:Number = 150;  
//bán kính c a chuy n ng tròn.  
var angleChange:Number = 10;  
//góc thay i giúp v t thay i góc quay và nó có th quay tròn  
var centerX:Number = stage.stageWidth/2;  
var centerY:Number = stage.stageHeight/2;  
//t a trung tâm mà v t s “bay” xung quanh  
var satellite:MovieClip = new Asteroid();  
satellite.x = satellite.y = -200;  
//t a c a v t khi b t u a vào flash  
addChild(satellite);  
  
addEventListener(Event.ENTER_FRAME, onLoop, false, 0, true);  
function onLoop(evt:Event):void{  
    var radian:Number = deg2rad(angle);  
    //chuy n giá tr c a góc “ ” sang radian  
    satellite.x = centerX + radius * Math.cos(radian);  
    satellite.y = centerY + radius * Math.sin(radian);  
    //t a m i c a v t  
    angle += angleChange;  
    //t ng góc  
    angle %= 360;  
    //khi angle t ng lên n 360 nó s chuy n v 0 ( m b o góc quay s không  
    + )  
}
```

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE

```
function deg2rad(deg:Number):Number {
    return deg * (Math.PI/180)
}
```

Rotation Toward an Object

(làm quay v m t phía nh t nh c a i t ng)

Chúng ta s d ng t a c am t v t nh h ng góc và l n.

D án c a chúng ta s trông nh sau:

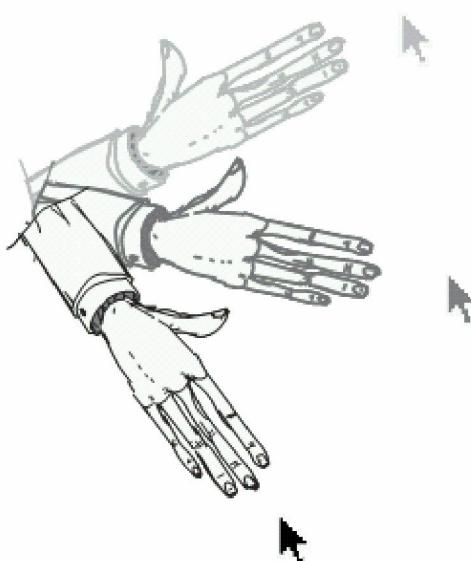


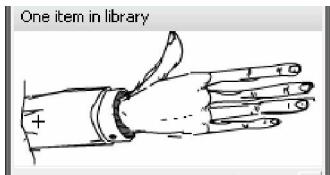
Figure 7-7. Using atan2(), you can continuously point a movie clip at the mouse no matter where it is on the stage

M t bàn tay quay theo con chu t và luôn xoay v phía con chu t.

Chúng ta s d ng hàm **atan2()**, chúng ta s có nó vì nó là m t ph ng pháp s n có c a l p **Math**. D a vào m t ch c n ng cho l p **Math** c a flash. Chúng ta xác nh góc gi a 2 i m, 2 i m trong 1 m t ph ng t o cho chúng ta 1 o n th ng mà góc chúng ta v a nói trên t c là góc h p b i o n th ng và tr c Ox, vì sao la tr c Ox?

n gi n v i n u oan th ng // v i tr c Ox thì góc h p b i chúng là 0 , i u này phù h p v i h góc c a flash.(xem l i hình 7-4).

B n c n ch c ch n r ng cánh tay ang ch sang bên trái // tr c Ox nó ch úng 0 trong flash, cánh tay ch a ho t ng và d nhiên thu c tính c a nó rotation c a nó c n là 0.



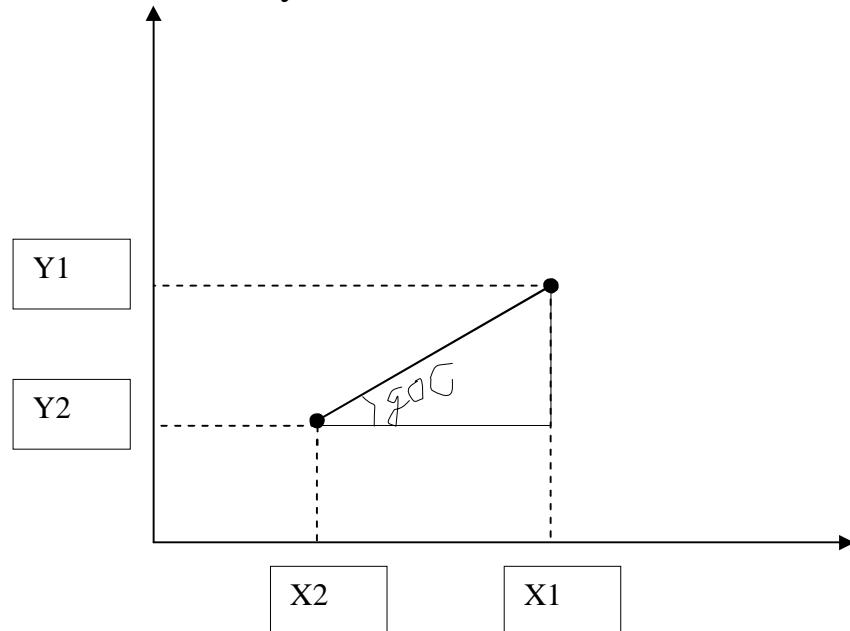
Câu lệnh trong chương trình này mà chúng ta cần chú ý là:

var radians:Number = Math.atan2(y1-y2, x1-x2);

bởi Math.atan2() cho chúng ta 1 hàm tan ngarc, nгин là $\tan 45^\circ = 1 \Rightarrow \tan^{-1}(\text{góc là tan ngarc}) = 45^\circ$.

tan mảng góc là tần số các nhánh của góc ở trên cạnh kề của góc ở trong 1 tam giác vuông (góc không phải là góc vuông của tam giác)

Trục Oy hình xuôi và trục Ox hình ngang sang pha, mà góc nằm tâm của góc cung tay (trong hình này góc tâm tay chỉ là tâm của cánh tay), **y1-y2** chỉ u lên trục Oy là cách còn **x1-x2** chỉ u lên trục Ox là cách kề.



```

var hand:MovieClip = new Hand();
hand.x = stage.stageWidth/2;
hand.y = stage.stageHeight/2;
// tay trung tâm màn hình
addChild(hand);
addEventListerner(Event.ENTER_FRAME, onLoop, false, 0, true);
function onLoop(evt:Event):void {
    hand.rotation = getAngle(hand.x, hand.y, mouseX, mouseY);
    //góc của bàn tay cung nhau bởi hàm getAngle()
}

```

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE J

```

function getAngle(x1:Number, y1:Number, x2:Number, y2:Number): Number {
    var radians:Number = Math.atan2(y1-y2, x1-x2);
    //tính tan ng c d a trên t a 2 i m. n v radian
    return rad2deg(radians);
    // i n v sang thu c tính rotation có th s d ng
}
function rad2deg(rad:Number):Number {
    //chuy n t n v radian sang
    return rad * (180/Math.PI);
}
```

Chúng ta s d ng VD này cho m t MovieClip n gi n, nh ng ng d ng c a nó r t l n, b n có th làm cho m i th xoay quanh m t i t ng, t o ra tính tr c quan và t ng tác r t có hi u qu trong ch ng trình.

Physics

(v t lý)

Y u t v t lý xu t hi n trong game, ho t hình, ho c trong nh ng thí nghi m n gi n, tuy nhiên v t lý có th nâng nh ng th k trên l ên m t t m cao m i, t o ra m t s thích thú m nh m v i ng i s d ng. Th m chí v i nh ng ng i giàu kinh nghi m, nh ng o n m ā nh n áng ng c nhiên c ng có th t o ra m t hi u ng v t lý áng k .

Chúng ta th o lu n v v t lý không có ngh a là chúng ta ang h c môn v t lý, chính xác thì chúng ta ch s d ng nh ng KQ v t lý ā bi t áp d ng vào ch ng trình c a chúng ta.

Các nhà khoa h c phát hi n ra các hi n t ng và quy v các công th c, chúng ta ch s d ng công th c ó m t cách n gi n nh t. Cái g i c ng liên quan n th c t , chúng ta n ên nhìn nh n v n này m t cách tích c c, N u chúng ta l p trình mà xa r i th c t thì c ng ch ng làm g i.

Ng n g n, th t có ích . tr c h t h ãy óng vai qu o n gi n c a m t hành tinh tr c khi xem xét s tr ng thái quay vòng c a m t hành tinh trên qu o, s h p d n c a l c h p d n t nh ng thiê n th kh ác, vân vân.

Gravity

(tr ng l c)

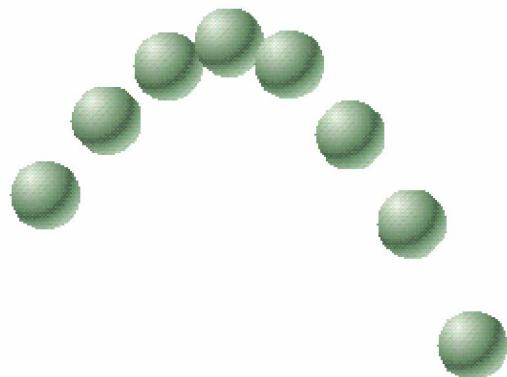


Figure 7-8. The effect of gravity on acceleration

Chuy n ng c a m t v t ch u tác d ng c a tr ng

l c

Chúng ta b t u v i hi n t ng v t lý quen thu c, ó là tr ng l c. Tr ng l c thì h ng xu ng, i u này ai c ng bi t, Flash cho phép chúng ta gi hi u ng c a tr ng l c m t cách tuy t v i, b ng cách thêm m t l c kéo nh tr c Oy (thêm t c là công thêm vào trục Oy m t vect gia t c tr ng tr ng m i c a tr ng l c), tr c Ox gi nguyên n gi n vì tr ng l c không tác d ng theo ph ng ngang :D.

```
var ball:MovieClip = new Ball();
ball.x = ball.y = 100;
addChild(ball);
var xVel:Number = 4;
var yVel:Number = -10;
var yAcc:Number = 1;
addEventListener(Event.ENTER_FRAME, onLoop, false, 0, true);
function onLoop(evt:Event):void {
    ball.x += xVel;
    ball.y += yVel;
    yVel += yAcc;
}
```

Chúng ta ã nói n gia t c ph n tr c, m i hành tinh u có m t gia t c tr ng tr ng nh t nh và nó là m t h ng s . VD trái t là $9,8 \text{ m/s}^2$. Chúng ta t o ra s khác bi t gi a các hành tinh v i nhau thông qua tr ng l c

Mã này có hi u qu cho vi c tung lên m t qu bóng vào trong khôn khí. Ban u qu bóng di chuy n ch m d n u vì ch u gia t c ng c, ang h ng lên khi gia t c

B N D C H NÀY C VI T B I HUYETSATHIENHA – thanh_vinh648@yahoo.com
12/1 – THPT QU C H CHU - B N D C H L U HÀNH TRÊN <http://WWW.VNFX.COM>

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE J
trong trang hổng xu ng. Sau khi t v trí cao nh t nó di chuy n nhanh d n u vì
cùng hổng v i chi u c a giat c. i u này phù h p v i các quy lu t c a v t lý.
Hãy tham gia trên website c a quy n sách này th y d án wall_bounce.fla, chúng
ta s thay i nhi u i u nh ranh gi i, gian o n, k t c u, th i gian n y v.v... c
b sung vào ch ng trình.

Friction

(ma sát)

Khi b n i trên ng, tr t trên sân b ng... t t c nh ng ho t ng ó ut o ra ma
sát.ng i ta có th làm cho Ma sát ng t ng c ng trên ng nh m m c ích
làm cho các xe tham gia giao thông có th d ng l i m t cách d dàng, ma sát c
gi m i trên các máy móc, thi t b b o m tu i th cho chúng. M t n g i n cách
thêm ma sát vào M t ho t c nh là t o ra M t h s ma sát mà d n d n gi m b t
v n t c trong c th i gian mà v t ch u ma sát ang chuy n ng.

Chúng ta s b t u t ví d c a ph n: “Movement Along an Angle”

thêm ma sát b n ch c n làm 2 vi c. u tiên t o ra m t h s ma sát (l n h n 0,
bé h n 1), sau ó nhân h s này v i giá tr c u v n t c ==> giá tr c a v n t c s
gi m d n theo th i gian, ó là i u chúng ta ch i.

```
var ball:MovieClip = new Ball();  
ball.x = ball.y = 100;  
addChild(ball);  
var speed:Number = 12;  
var angle:Number = 45;  
var radians:Number = deg2rad(angle);  
var xVel:Number = Math.cos(radians) * speed;  
var yVel:Number = Math.sin(radians) * speed;  
var frCoeef:Number = .95;  
//thêm h s ma sát  
addEventListener(Event.ENTER_FRAME, onLoop, false, 0, true);  
function onLoop(evt:Event):void {  
    xVel *= frCoeef;  
    yVel *= frCoeef;  
    //v n t c gi m do b ma sát  
    ball.x += xVel;  
    ball.y += yVel;  
}  
function deg2rad(deg:Number):Number {
```

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE J

```

return deg * (Math.PI/180);
//chuyen n n v t sang radian
}

```

Zeno’s Paradox

(ngh ch lý c a zeno)

Cách khác thêm ma sát vào s chuy n ng i t ng s s d ng ngh ch lý Zeno , mà nói i u này, khi di chuy n t i m này sang i m khác, b n không bao gi th t s t n n i n t n cùng i m s n c a b n b i vì b n s chia c t ng i Còn l i (kho ng cách) v i m i s chuy n ng có ma sát.

Sách nó vi t v y thoai, có ngh a là b c u tiên i t a => b c n 100 cm chúng ta b t l i còn 50cm, ti p theo muôn i n b c n 50 cm n a, chúng ta b t l i nó ch i c 25 cmm và c th , nó có th s khong n c n i c n n, d i khái ni m c a toán h c thì gi i h n c a vi c gi m b t này cu i cùng v n là n ích thành công, tuy nhiên i u này x y ra khi s b c di chuy n là , t c là s bu c di chuy n r t nhi u, tuy nhiên ó là toán h c cong th c t trong flash thì nó l i khác, vì s 1 n l p l i trong EventListener c a ENTER_FRAME là r t l n nê b c di chuy n c a qu bóng là r t l n, m t khác do kick th c qu bóng to và kho ng cách di chuy n h p nê ta khong th y rō i u này.

N u b n chia kho ng cách t i m a n i m b b ng m t n a b c b n s khong bao gi n c i m b ý t ng này có th s d ng làm ch m m t i t ng trong khi nó ti p c n n i n ích c a nó, nh c minh h a trong Hình 7 9



Figure 7-9. Zeno’s paradox, a simple way to depict friction or easing

Mã l nh c a ch ng trình nh sau:

```

var ball:MovieClip = new Ball();
ball.x = ball.y = 100;
addChild(ball);
addEventListerner(Event.ENTER_FRAME, onLoop, false, 0, true);
function onLoop(evt:Event):void {
    ball.x += velFriction(ball.x, mouseX, 8);
    ball.y += velFriction(ball.y, mouseY, 8);
}
function velFriction(orig:Number, dest:Number, coeff:Number):Number {

```

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE J
//nguyên lý mình đã trình bày, cái khác là rằng trình này nó chia cho 8 ch không ph i 2

return (dest-orig)/coeff;

}

VD: khoảng cách giữa ball và MOUSE là 512 -> ta sau đó c u tiên là bóng là $x + 64$, => khoảng cách còn lại giữa MOUSE và bóng là $512 - 64$, b c tiếp theo ball di chuyển n m t b c là $(512 - 64)/8$ khoảng cách l i g n l i và c th ...

Có thể xem coeff như là m t d ng h s ma sát, khi coeff càng cao thì chuyển n ng càng chậm, m t nhanh ut ng gian h n và ng c l i.

Elasticity

(s àn h i)

Một tính chất của vật lý có thể ứng dụng vào học là s àn h i, thu tính này có thể áp dụng cho các mô hình chuyển động khác nhau và có thể có nhiều cách khác nhau để ứng dụng nó. Lực àn h i xuất hiện khi có một vật tiếp xúc với 1 lực nào đó bị biến dạng và lực àn h i xuất hiện kéo vật trở lại vị trí, trạng thái ban đầu không bị nguyên nhân biến dạng đó

Điểm gốc của các công thức vật lý thì lực àn h i tính bằng $\mathbf{F} = -k\mathbf{x}$

$F = -k(x - x_0)$

$k = \frac{F}{x}$

Lực này chỉ xảy ra biên dạng của vật nên có dấu “-”.



Figure 7-10. A basic depiction of easing using Hooke's law of elasticity

VD trên mỗi lần cập nhật gi i quy t m t MovieClip trong nh ng vị trí nh ng khu vực khác nhau. Hình vẽ miêu tả 1 quả bóng, Chuyển động về phía con chuột (có dấu +), ban đầu nó chuyển động quanh con chuột, sau đó nó cân bằng tại vị trí con chuột.

```
var ball:MovieClip = new Ball();
ball.x = ball.y = 100;
addChild(ball);
var xVel:Number = 0;
var yVel:Number = 0;
```

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE J

```

addEventListener(Event.ENTER_FRAME, onLoop, false, 0, true);
function onLoop(evt:Event):void {
    xVel = velElastic(ball.x, mouseX, .14, .85, xVel);
    yVel = velElastic(ball.y, mouseY, .14, .85, yVel);
    ball.x += xVel;
    ball.y += yVel;
}

```

Chúng ta xem xét tiếp hàm **velElastic()**

```

function velElastic(orig:Number, dest:Number,
springConst:Number,damp:Number, elas:Number):Number {
    elas += -springConst * (orig - dest);
    return elas *= damp;
}

```

Thì còn lối chính là viết i tính lối àn hối, biến springConst để xem nhữn g k (càng cao thì lối àn hối càng mạnh) còn (orig - dest) để xem nhữn g khoảng cách cần di chuyển, dù trục t cùn chuy n ng có thay đổi ngay khi vật chuy n quá giới hạn àn hối, chỉ ban đầu orig < dest (tất cả vật bé hơn ta mà nó sẽ chuy n từ i) => orig - dest < 0 => -springConst * (orig - dest) > 0 => vật ti n lên trục, sau đó vật sẽ qua vị trí tiếp theo orig > dest => -springConst * (orig - dest) < 0 và chuy n về lui.

elas += elas += -springConst * (orig - dest);

câu lệnh này tạo cho vật chuy n với vận tốc **elas += -springConst * (orig - dest)** biến thiên (biến thiên nhữn g nào thì mình nói rõ)

Nếu chia động lượng nhữn g ấy thì vật sẽ chuy n qua lối quanh con chuỗi không lối đi và không yên t i con chuỗi, chúng ta nhận thấy vận tốc 0 < **damp** < 1, vật sẽ tiếp tục lối đi và không yên t i con chuỗi.

//sách nó không viết cách công thức mình viết đâu, cái này là mình cung cấp và //viết rõ).

Programmatic Tweening

(lối trình cho tween)

Khi bạn cần một hình ảnh gián và không muốn mất thời gian công sức viết mã lối, bạn có thể sử dụng cáchTween s trong có của flash, tạo nó trên timeline. Một ví dụ là file as.Tween.fla kèm theo.

Lối tween sẵn có trong AS3 cho phép bạn chỉ rõ xem vật là i tay ngón tay (cho chuy n tay) và tween như thay nào, sử dụng mã lối tạo tween cho phép

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER 'S GUIDE J
thay i thu c tính, giá tr b t u và hoàn thành c a thu c tính, kho ng th i gian C a tween, Và, Cu i cùng, Nên ch ng s d ng th i gian ho c nh ng khung (frame) khi vi c c l ng tween có trong 1 kho ng th i gian nh t nh.

ây là c u trúc c a hàm Tween:

Tween(obj:Object, prop:String, func:Function, begin:Number, finish:Number, duration:Number, useSeconds:Boolean)

Chúng ta có th t o nhi u hàm cho 1 i t ng, hay s d ng m t hàm cho nhi u i t ng t c hi u qu mong mu n

Obj:tên i t ng

Prop; thu c tính tham gia tween (x, y)

Func: hàm thêm vào (VD Elastic.easeOut chuy n ng ch m d n trong m t hi u ng àn h i...)

Begin: giá tr ban u tr c khi tween.

Finish: giá tr sau khi tween.

Duration: th i gian th c hi n(ch a có n v)

useSeconds: thi t l p true hàm s d ng n v giây tính th i gian **Duration**.

VD sau cho chung ta t o 1 tween cho m t i t ng c t o ng trên màn hình có chuy n ng châm d n, do s d ng hàm **Elastic.easeOut**:

```
import fl.transitions.Tween;
import fl.transitions.easing.*;
var ball:MovieClip = new Ball();
ball.x = ball.y = 100;
addChild(ball);
var ballXTween:Tween = new Tween(ball, "x", Elastic.easeOut, 100, 400, 3,
true);
//cho thu c tính x
```

Chúng ta vi t thêm hàm cho thu c tính y và alpha:

```
var ballXTween:Tween = new Tween(ball, "x", Elastic.easeOut, 100, 400, 3,
true);
var ballAlphaTween:Tween = new Tween(ball, "alpha", None.easeOut, .3, 1, 3,
true);
// None.easeOut có ngh a là không s d ng ch c n ng easeOut ch m d n trong tween c a //alpha.
```

L p Tween có vài thu c tính b sung, nh ng ph ng pháp, và nh ng s ki n cho s s d ng b i s th hi n c a m i l p. áng chú ý là các thu c tính boolean cho vi c ch y (playing) và l p l i c a tween (looping) (ch s d ng cho ho t c nh khi ang

Table 7-1. Easing types found in the fl.transitions.easing package

Easing Class	Description
Back	Easing in begins by backing up and then moving toward the target. Easing out overshoots the target and backtracks to approach it.
Bounce	Bounces in with increasing speed, or out with decreasing speed.
Elastic	Undulates in an exponentially decaying sine wave, accelerating in and decelerating out.
None	Linear motion without easing.
Regular	Normal easing, like that found in the timeline's simple easing feature, accelerating in and decelerating out.
Strong	Emphasized easing, stronger than that found in the timeline's simple easing feature, but without additional effects. Accelerates in and decelerates out.

BIÊN SO NL IT SÁCH
 ch y và l p) và s v trí
Number position,
 thu c tính v trí ch báºo
 giá tr hi n th i c a
 thu c tính tweening,
 nh v y nó tham chi u
 t i v trí hi n th i c a
 tween, khôºng ph i là v
 trí X /Y trên màn hình
 ch a i t ng trên

.Cái ó th hi n ballAlphaTween xem tr c ó v n còn báo l i bi n v trí, m c dù mà
 giá tr alpha c a MovieClip l i ang c tween

the current value of the tweening property, so it refers to the current position
 of the tween, not the x/y position of the display object on stage—that is, the
 ballAlphaTween instance seen previously still reports the position variable,
 even though the alpha value of the movie clip is being tweened

v i gói fl.transitions.easing khi import vào flash, chúng ta có các ph ng pháp sau
 th c hi n:

Back: Vi c làm m m chuy n ng b t ub i l u l i và sau ó chuy n ng v phia
 ích.Vi c gi m nh ngoái,khi v t chuy n ng quá ích và nó rút ng c tr l i
 ti p c n ích

Bounce: Nh ng s N y ho c là gi t bên trong v i t c ngày càng t ng, ho c
 ngoái v i vi c gi m b t t c .

Elastic: àn h i g n sóng, xây d ng chuy n ng theo dao ng hình sin, t ng t c
 bên trong và gi m t c ngoái.

V n Ch h ng chuy n ng ch khôºng có hi u ng gi m nh

Regular: Làm m m m t cách Bình th ng, gi ng nh cách làm m m chuy n ng
 trên timeline, t ng t c bên trong và gi m t c ngoái.

Strong: Nh n m nh làm m m, chuy n ng m nh m h n so cách làm trên
 timeline làm m m c tính, nh ng khôºng có b sung nh ng hi u ng khác (gi m nh - làm m m). T ng t c bên trong và gi m t c ngoái

M i l p cho phép ít nh t ba ph ng pháp bao trùm làm m m bên trong, làm m m
 ngoái, và làm m m c hai trong khi k t thúc tween

T t c các ph ng pháp cho m i l p h tr b i nh ng giá tr chung cu c và ban u
 c a thu c tính khi b t u ho t hình, kho ng th i gian làm m m , và th i gian hi n
 t i trong ho t c nh khi ang ch y.

B ND CH NÀY C VI TB I HUYETSATHIENHA – thanh_vinh648@yahoo.com
12/1 – THPT QU CH CHU - B ND CH L U HÀNH TRÊN <http://WWW.VNFX.COM>

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE

Sau c ng h tr m t giá tr cho chuy n ng quá m c v tra bên ngoài ích t n oi b t u và nn i k t thúc c a ho t c nh, b sung l c àn h i h tr cho biên và chuy n ng sóng th hình sin, ph ng pháp ã tính toán s àn h i.

Timeline Animation Recreations

(t o ho t hình t timeline b ng mă l nh t ng)

Trong khi ây không th c s tr n v n là m t gi i pháp dùng ActionScript, chúng ta thích dùng flash th c hi n ho t hình

CS3 có nh ng l p Chuy n ng và công c ho t hình (**Animator**) m i.Nh ng l p này, và nh ng h tr ng i dùng m i c a CS3, làm cho nó là có th ch y l i nh ng ho t hình mà ã c t o ra tr c ó trong timeline.

Nh ng ng i theo ch ngh a us d ng Scripting có th quan tâm h n n vi c làm nâng cao k n ng ActionScript c a h h n là s d ng timeline b t ngu n nh ng ho t c nh.Tuy nhiên Kh n ng s d ng mă l nh c a flash h p d n nhi u ng i n v i flash CS3 h n , th m chí dùng code curmudgeons. cho 2 lý do:

However, this capability may be attractive to many Flash CS3 users, even coding curmudgeons, for two reasons

Tr c h t, nó không s d ng timeline nh ng v n còn làm có th tái t o l i ho t hình ra ó - bao g m nh ng ho t c nh mà cái ó có th không là d ó t c chính xác nh v i vi c s d ng ActionScript.

First, it doesn’t ultimately use the timeline but still makes it possible to reproduce complex animfations created there—including animations that might not be that easy to achieve strictly with ActionScript

Th 2 nó là m t cách m i liên k t h p tác gi a l p trình viên và designer, Nh ng ng i thi t k có th t o ra nh ng ho t c nh timeline, và nh ng ng i l p trình có th c l p làm vi c vào trong nhi u d án khác nhau mà không ph thu c c u trúc b n chính timeline.

Second, it offers a new path to improved designer-programmer collaboration, Designers can create timeline animations, and programmers can integrate that work into other projects without relying on the original timeline structure

N n t ng c a quá trình này t n t i trong m t c tính g i là “Copy Motion as ActionScript 3.0.” Sau vi c t o ra m t timeline tween, B n có th l a ch n toàn b tween và sau ó ch n “Copy Motion as ActionScript 3.0.” t th c n Edit =>Timeline.Nh ng s sao chép này t i t i clipboard, cho nó có có t t c Thông tin c n thi t t o l i tween v i mă l nh. Trong th i gian quá trình sao chép, c tính nh c b n t tên th hi n (instance name) ký hi u tweened v i m t h i tho i ti n l i – s báol i n u tên th hi n ã t n t i. Khi quá trình copy ã hoàn thành, b n có th dán

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE

Kết quả vào **Actions panel** (nút bấm làm theo các bước mình hướng dẫn, dán bùn chìm Ctrl + V, thà là xong, nó sẽ hiển thị ra mã XML, i khái gì nghe thắc này:

```

import fl.motion.Animator;
var ball_xml:XML = <Motion duration="20" xmlns="fl.motion.*"
xmlns:
geom="flash.geom.*" xmlns:filters="flash.filters.*">
<source>
    <Source frameRate="12" x="50" y="50" scaleX="1"
scaleY="1"
        rotation="0" elementType="movie clip"
instanceName="ball"
        symbolName="Ball">
        <dimensions>
            <geom:Rectangle left="-10" top="-10"
width="20"
                height="20"/>
        </dimensions>
        <transformationPoint>
            <geom:Point x="0.5" y="0.5"/>
        </transformationPoint>
    </Source>
</source>
<Keyframe index="0">
    <tweens>
        <SimpleEase ease="0"/>
    </tweens>
</Keyframe>
<Keyframe index="19" x="450"/>
</Motion>;

```

Tất cả ActionScript có bao gồm chuyển động có thể được sao chép và粘贴 tin XML trong định dạng (format) yêu cầu bao gồm Chuyển động (motion class). Mình ví dụ như sau, (mà) tweening (mà) Một movieclip có thời gian hoạt động là 20 frame, như sau:

The foundation of this process exists in a feature called “Copy Motion as ActionScript 3.0.” After creating a timeline tween, you can select the entire tween and then choose the Copy Motion as ActionScript 3.0 menu option from the Edit®Timeline menu. This copies to the clipboard all necessary information to recreate the tween with code. During the copy process, the fea-

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE J
ture prompts you for the tweened symbol’s instance name with a convenient dialog—prepopulated if the instance name already exists.

Once the copy completes, you can paste the results in the Actions panel. All the ActionScript needed is included, and the motion is represented by XML information in the format required by the Motion class. A simple example, tweening a movie clip across the stage over 20 frames, follows:

```
import fl.motion.Animator;
var ball_xml:XML = <Motion duration="20" xmlns="fl.motion.*" xmlns:
geom="flash.geom.*" xmlns:filters="flash.filters.*">
<source>
  <Source frameRate="12" x="50" y="50" scaleX="1" scaleY="1"
  rotation="0" elementType="movie clip" instanceName="ball"
  symbolName="Ball">
    <dimensions>
      <geom:Rectangle left="-10" top="-10" width="20"
      height="20"/>
    </dimensions>
    <transformationPoint>
      <geom:Point x="0.5" y="0.5"/>
    </transformationPoint>
  </Source>
</source>
<Keyframe index="0">
  <tweens>
    <SimpleEase ease="0"/>
  </tweens>
</Keyframe>
<Keyframe index="19" x="450"/>
</Motion>;
//th là xong ph n XML, chúng ta không th o lu n v XML ày
var ball_animator:Animator = new Animator(ball_xml,
ball);
//t o m t tween b ng mă l nh d ng s n c a AS3
ball_animator.play();
//bibi nào :D
```

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER 'S GUIDE

B n có th vi t riêng m t file *.xml b ng cách s d ng oan code b t u t th <Motion> và k t thúc b ng th </Motion> c th :

```
<Motion duration="20" xmlns="fl.motion.*" xmlns:geom="flash.geom.*" xmlns:filters="flash.filters.*">
<source>
<Source frameRate="12" x="50" y="50" scaleX="1" scaleY="1"
rotation="0" elementType="movie clip" instanceName="ball"
symbolName="Ball">
<dimensions>
<geom:Rectangle left="-10" top="-10" width="20"
height="20"/>
</dimensions>
<transformationPoint>
<geom:Point x="0.5" y="0.5"/>
</transformationPoint>
</Source>
</source>
<Keyframe index="0">
<tweens>
<SimpleEase ease="0"/>
</tweens>
</Keyframe>
<Keyframe index="19" x="450"/>
</Motion>
```

D án chúng ta s p nghiên c u sau ây là m t ng d ng t mă l nh làm tween t ng c a AS:

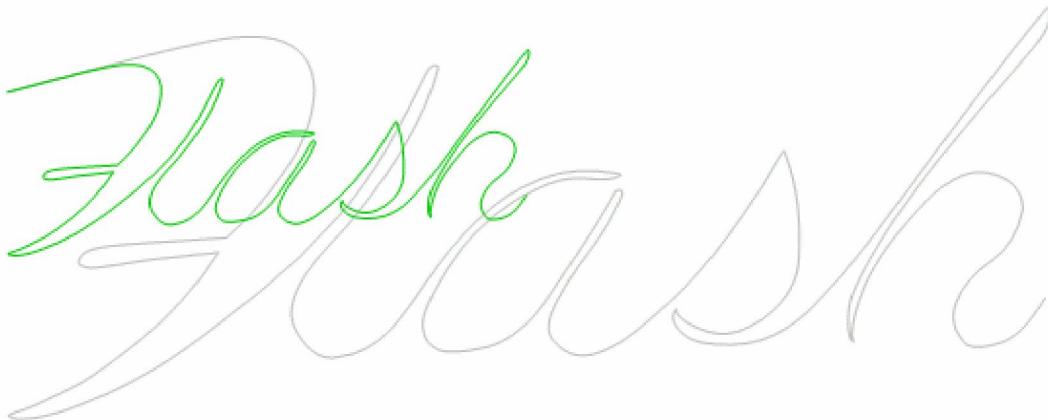


Figure 7-11. The motion guides used to create a tween that is recreated with the Animator class

Chúng ta tìm hiểu mảng tý và hàm flash.geom

Gói flash.geom chứa các đối tượng hình học, là những hình ảnh, những ma trận hình ảnh và biến số, hỗ trợ lớp BitmapData và các tính bitmap caching.

Hình ảnh có thể hiển thị có 1 quỹ đạo di chuyển theo hình vẽ (ngày màu đen, và màu xanh), tham khảo file mẫu xem hướng dẫn và các tài nguyên sẵn có trong ó.

Ví dụ này tạo ra một player có thể phát video, có nút điều khiển ban đầu trong một file mẫu. Nó không yêu cầu cái gì Hỗn hợp movie clip làm hình, Và Tạo nút bấm thành phần, Nào Chúng ta sẽ có tạo ra nút có chức năng dừng khi nhấp vào hình. 8 hàng đầu tiên của mã lệnh là những gói công thức, khai báo những biến C# và tạo ra và nhúng vào một quỹ đạo movieclip để thêm nó vào danh sách màn hình.

```
import fl.motion.*;
import flash.geom.*;
import fl.controls.Button;
//gói tạo nút bấm
var anim:Animator;
//biến hỗ trợ hình ảnh AS3
var isPaused:Boolean;
//có nút "paused" hay không khi đang chạy chương trình
var isScaled:Boolean;
//Có sử dụng Chuyển động mượt (nhlyn) hay không
```

BIÊN SO N L I T SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE

```
var ball:Sprite = new Ball();
```

```
ball.x = ball.y = 80;
```

Chúng ta liên kết file XML, cách tạo XML trong flash và tạo một file XML riêng mình ở trình bày, chúng ta đi sâu vào XML trong Chapter 13:

```
var xml_url:URLRequest = new URLRequest("handwriting.xml");
```

```
var xml_loader:URLLoader = new URLLoader(xml_url);
```

```
xml_loader.addEventListener("complete", xmlLoaded, false, 0, true);
```

Hàm **xmlLoaded()** chuyển đổi thành phần XML và **Animator class**, qua XML làm thế nào movieclip quay bóng cho lặp. Tuy nhiên, việc chuyển động có thể cung cấp thông tin qua **XML** mà không cần tách.

```
function xmlLoaded(evt:Event):void {
```

```
    var anim_xml:XML = XML(xml_loader.data);
```

```
    anim = new Animator(anim_xml, ball);
```

//Sau đó XML sẵn có, chúng ta bắt đầu

```
    ball.transform.colorTransform = anim.motion.keyframes[0].color;
```

//tạo màu chuyển cho ball tại keyframe đầu tiên (t =>xanh, vì sao thì chưa rõ) **addChild(ball);**

```
    anim.addEventListener(MotionEvent.MOTION_END, onMotionEnd, false, 0, true);
```

//MotionEvent.MOTION_END là sự kiện listener kéo dài ra khi MOTION kết thúc

```
}
```

```
function onMotionEnd(evt:MotionEvent):void {
```

```
    Button(getChildByName("Play")).label = "Play";
```

//đặt tên nút thành play khi kết thúc MOTION

//chúng ta tìm hiểu về nút bấm phím sau, còn vì sao kết thúc motion mà phải chuyển tên //chúng ta đang sử dụng sau

```
}
```

Sau đây là mã lệnh tạo nút bấm có flash thông qua button component

```
createController(["Play","Pause","Stop","Next Frame","Toggle Scale"]);
```

//tạo 5 nút và đặt tên cho các nút theo mặc định bình thường

```
function createController(bttns:Array):void {
```

```
    for (var i:Number = 0; i<bttns.length; i++) {
```

//vòng lặp for tạo tên, nhãn (label) cho từng nút

//tên là tên của nút chúng ta thực hiện tác khi dùng mã lệnh

//label là cái mà người dùng nhìn thấy, chỉ liên quan đến việc click vào khi nhấn

```
        var btn:Button = new Button();
```

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER 'S GUIDE

```
btn.x = 35 + i*100;
btn.y = 350;
btn.width = 80;
btn.label = btns[i];
btn.name = btns[i];
btn.addEventListener(MouseEvent.CLICK, onNav, false, 0, true);
//g i hàm th c hiên onNav() trong m i EventListener thích h p cho m i nút b m
khác //nhau
    addChild(btn);
}
}
```

D i ây là mã l nh cho t ng nút l nh khác nhau v a t o, mìn trình bày gi i thích
khá chi ti t trong mã l nh, mong các b n chú ý.

```
function onNav(evt:MouseEvent):void {
switch (evt.target.name) {
case "Play" :
//mã l nh cho nút play
if (!anim.isPlaying) {
//n u ho t hình ang không ch y nh n nút play k t h p thêm v i i u ki n d i
ây
if (isPaused) {
//isPaused m c nh khi khai báo là false, các câu l nh d i ây khong x y
ra
//N u nút Paused à nh n,ispause mang giá tr true (ho t hình à ng yên
//tr c khi nh n nút play) nút play cho phép ho t hình ch y l i
anim.resume();
//chuy n ho t hình ch y ti p
isPaused = false;
//chuy n giá tr sang ch a nh n nh n
//n u nút paused ch a nh n ho t hình v n ti p t c ch y nh bình th ng
//cau lệnh if cho tr ng h p isPaused là true này k t thúc t i ây
} else {
//n u isPaused ch a nh n
anim.play();
//ho t hình v n bibi nh bình th ng
}}
```

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE J

```

//sau khi nh n paused, ho t hình d ng l i và ch ch y l i khi nút “resume”
    c //nh n, nút resume ch là tên khác c a nút “play”
//cau lenh if cho tr ng h p isPaused là false này k t thúc t i ây

}

}

//sau khi nh n nút play, chuy n ng s b t u và ...
break;
case "Pause" :
//mã l nh cho nút paused
anim.pause();
//d ng ho t hình
isPaused = true;
//giá tr isPaused chuy n sang true
ButtongetChildByName("Play").label = "Resume";
// i tên play thanh resume, mãi nh th n cu i motion m i i l i thành
//play, bây gi thi b n hi u câu l nh i tên nút m c tr c r i ch
break;
//Nút paused khi c nh n làm ho t hình d ng l i và i tên nút play
case "Stop" :

anim.stop();
//d ng ho t hình
anim.rewind();
//ho t hình tr l i i m ban u
isPaused = false;
//nút paused xem nh ch a nh n, nh c
ButtongetChildByName("Play").label = "Play";
// i tên nút có tên Play sang label m i c a nó có tên “play”
break;
//Nút Stop khi c nh n s a m i th tr v v trí ban u.
case "Next Frame" :
anim.nextFrame();
//t i frame ti p theo c a ho t hình
break;

```

case "Toggle Scale" :

```
//n u nút Toggle Scale ã nh n các câu lệnh d i ây x y ra.  
var m:Matrix = anim.positionMatrix = new Matrix();  
//t o ma trân m theo ho t hình anim (t c là t o m t ho t hình m i gi ng anim  
var s:Number;  
if (isScaled) {  
//giá tr isScaled m c nh khi khai báo là true, các câu lệnh d i ây s  
ch y  
s = 1;  
//s là t s thu g n, s= 1 ngh a là s không có thu g n v ng d n  
isScaled = false;  
//giá tr i thành false câu lệnh d i ây x y ra,  
} else {  
s = .5;  
//t s thu g n s c thu g n l i-gi m i m t n a  
isScaled = true;  
};  
MatrixTransformer.setScaleX(m, s);  
//Chuy n ng theo chi u X gi m i m t n a (s = 0.5) ho c không gi m (s =  
1)  
MatrixTransformer.setScaleY(m, s);  
//Chuy n ng theo chi u Y gi m i m t n a (s = 0.5) ho c không gi m (s =  
1)  
break;  
}  
}
```

//Qua các câu lệnh, khi nh n nút Toggle Scale chúng ta s có m t qu o
b ng //m t n a qu o ban u

B n s d ng Matrix s giúp t o mã lệnh chuy n ng m t cách n gi n,
chúng ta ch nh s c m t s thu c tính c a Matrix thông qua hàm

MatrixTransformer.setScaleY(m, s); và MatrixTransformer.setScaleX(m, s);

Khi ó, m là m t chuy n ng m i c t o r a t mã l nh và nó i u khi n
cho MovieClip Ball chuy n ng.

VD này khó hi u, mình ã m t nhi u th i gian nghiên c u v nó, các b n
không hi u có th l ên forum ho c mail qua thanh_vinh648@yahoo.com

Particle Systems

(phân vùng h th ng)

Chúng ta s t o ra 1 h th ng khá p m t:



Figure 7-12. A particle system with a gravity setting of 2 and .2

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE J

Phân vùng h th ng là m t cách óng vai nh ng i t ng ph c t p ho c tài nguyên mà c bao g m n hi u Ph n nh , nh nh ng ch t l ng, pháo hoa, n tung, khai h a, khói, n c, tuy t, vân vân.

Nh ng h th ng Ph c t p thì tính kh thi th c hi n t ng ph n riêng l c phát sinh, c cho là nh ng c tr ng c a b n thân d i t ng, và chúng s t ch y m t cách t ng.

H n n a, nh ng ph n nh Thì d bi n i, s a i i u ch nh, ho c ang th m chí thay th , làm cho nó là có th thay i s xu t hi n ho c tính ho t ng c a h th ng l n m t cách t ng i d dàng

Nh ng i u óc ng là nh ng c tr ng c a l p trình h ng i t ng, vì v y k thu t này không có gì áng ng c nhiên vì r ng s d ng cách ti p c n này s dùng vi t phân vùng h th ng

T i k t thúc ch ng này, chúng ta t o ra m t h th ng t ng ph n r t n gi n - s d ng ch hai l p - mà nhìn m t s movieclip nh bé n gian t o thành m t o giác gi ng hi u ng n c su i (mình th y nó gi ng vời phun n c h n).

Màu c a nh ng vòng tròn l n ngoài C a “Su i” Và sau ó r i xu ng d i hi u ng c a s c n ng bao g m tr ng l c. Hình 7 12 cho chúng ta th y h th ng trông nh cái gì.

Sau vi c khai báo nh ng bi n xác nh nh ng i m mà ph n s xu t hi n, t t c l p c n thi t xây d ng ch ng trình t o Thêm M t Vào Eventlistener m t khung cho t i Stage.

trên M i Vào khung c a s ki n ENTER_FRAME, hàm listener t o ra m t Ph n l p m i c a h th ng (m t MC ball m i), mà t o ra m t Ph n c a nó và thêm nó vào danh sách màn hình.

package {

```
import flash.display.Sprite;
import flash.events.Event
public class ParticleDemo extends Sprite {
    private var emitterX:Number = stage.stageWidth/2;
    private var emitterY:Number = stage.stageHeight/2;
    public function ParticleDemo() {
        stage.addEventListener(Event.ENTER_FRAME, onLoop,
            false, 0, true);
    }
}
```

```
private function onLoop(evt:Event):void {
    var p:Particle = new Particle(emitterX,
```

```
    emitterY,  
    Math.random()*11 - 6,  
    Math.random()*-20,  
    1,  
    Math.random()*0xFFFFF);  
    addChild(p);  
}  
}  
}
```

File ngu n c a chúng ta là ParticleDemo.as

Mã l nh trên không có gì xa l v i chúng ta, ó là nh ngh a m t l p trong m t gói. Chúng ta có l p ParticleDemo.as có nhi m v t o các MovieClip có tên là p vào màn hình, vi c t o này là liên t c b i l nó t trong Event Listener ENTER_FRAME. p có nh ng c tính ã khai báo tr c khi s d ng hàm **Particle** chúng ta khai báo nh ng c tính c a p thông qua câu l nh var. Chúng ta s d ng các phân h th ng thành nh ng ph n nh (m i ph n là m t MovieClip Ball- t c t o ra, t ho t ng và t m t i m t cách t ng, ch ng trình l n không c n ph i s lý gì mà toàn b do hàm **Particle** qu n lý. nh ng qu bóng s d ng hàm **Particle** và hàm này có nh ng thu c tính hay ph ng pháp gì thì chúng ta hãy xem ti p ND c a hàm này:

package {

```
    import flash.display.*;  
    import flash.geom.*;  
    import flash.events.Event;  
    public class Particle extends Sprite {  
        //các giá tr c a t ng ph n nh trong h th ng chính là Particle  
        private var _xpos:Number;  
        private var _ypos:Number;  
        //t a x và y  
        private var _xvel:Number;  
        private var _yvel:Number;  
        //t c c a x và y  
        private var _grav:Number;  
        //tr ng l c  
        public function Particle(xp:Number, yp:Number, xvel:Number, yvel:Number,  
        grav:Number, col:uint) {
```

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE //hàm nh p d li u t l p DemoParticle truy n qua, bao g m các thu c tính c a Particle

```
_xpos = xp;
_ypos = yp;
_xvel = xvel
_yvel = yvel
_grav = grav;
var ball:Sprite = new Ball();
addChild(ball);
x = _xpos;
y = _ypos;
// nh v trí, b t k , b t k nh th nào thì xem r ng các hàm Math.random c a l p
DemoParticle
alpha = .8;
//gi m m c
scaleX = scaleY = Math.random() * 1.9 + .1;
//Kick th c ngûu nhiên
var colorInfo:ColorTransform = ball.transform.colorTransform;
//khai báo bi n màu
colorInfo.color = uint(col);
//Giá tr màu c a ball là bi n col mang giá tr ng u nhiên, xem l i l p DemoParticle s
th y
ball.transform.colorTransform = colorInfo;
//màu trái banh quy nh t bi n colorInfo có màu ng u nhiên l y t bi n
colorInfo.color
addEventListener(Event.ENTER_FRAME, onRun, false, 0,
true);
}
private function onRun(evt:Event):void {
_yvel += _grav;
//tr ng l c h ng xu ng
_xpos += _xvel;
_ypos += _yvel;
x = _xpos;
y = _ypos;
//Qu bóng di chuy n
```

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER 'S GUIDE ↴

```
if (xpos < 0 || ypos < 0 || _xpos >
    stage.stageWidth || ypos > stage.stageHeight) {
//Nếu quay bóng ra khỏi màn hình hoặc (có tác động quá kick của màn hình) thì :
    removeEventListener(Event.ENTER_FRAME, onRun);
    parent.removeChild(this);
//Xóa bỏ hàm onRun và xóa luôn quả bóng đó ra khỏi danh sách màn hình.
}
}
}
```

Chúng ta thấy rõ ràng: Vì cách chia thành các phần nhỏ giúp chương trình truy cập nhanh và linh hoạt hơn không cần thông qua một lớp, chúng ta truy cập dữ liệu cho một lớp khác dễ dàng, và cách truy cập (thực hiện không có mã lệnh chương trình chính) và chương trình xử lý theo mã lệnh đã ghi trong lớp xử lý dữ liệu. Vì cách chia nhỏ này giúp chúng ta dễ dàng thay đổi cách hiển thị và bổ sung một cách nhanh chóng là một cách trong cách OOP thông qua việc sử dụng lớp.

Thay đổi các thuộc tính alpha, scaleX và scaleY, chúng ta sẽ có những hiệu ứng khác nhau. Nhìn ví dụ, chúng ta có thể obranh phiên bản khác của chương trình này trên web-site của quyển sách mà bao gồm vài thuộc tính mới, bao gồm kiểuуль và hình ảnh để tra thêm mà bạn sử dụng chúng trong chương trình theo.

Xong chapter 7 một quá L các bạn nhé thanks! Hãy tinh thắn cho mình nha!

CHAPTER 8 Drawing with Vectors

Biên soạn bởi huyetsathienha đã trên sách Learning.Action.Script.Beginner 's guide
Thanh_vinh648@yahoo.com

Chương này chúng ta sẽ bàn luận về các vấn đề sau:

The Graphics class: Lớp này, tình cờ mà flash sử dụng một thang toán học API để trên vector. Bạn có thể hiểu khi viết toán học và vẽ, hay tô màu

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE J
vùng c v b ng nh ng hàng c n i t nh ng i m mà chúng ta d ng s n b ng
mã l nh, t ng t ng chúng ta s d ng m t cây bút vô hình làm vi c ó
The Geometry package: Gói hình h c. Gói có ích này ch a ng nh ng l p cho t o
ra nh ng i m và nh ng hình ch nh t, c ng nh thay i nh ng i t ng và màu
c a nó, và vi c t o ra ma tr n toán h c cho ng th i ph c t p thay i t i thu c tính
góc (rotation), bi n i b r ng ho c v ngang (scale), Và chuy n i theo X và Y.
9-slice Scaling: . B qua s s d ng c a m t hình ch nh t có th ch nh b ng mã
l nh, Chúng ta s d ng 9-slice Scaling ng n ng a nh ng c nh và nh ng góc c a
m t sprite ho c movie clip l i T s thay i hình d ng (distorting) Khi thay i chi u
r ng và chi u ngang (scale – thay i t l c a i t ng)

Applied Examples: (áp d ng ví d): K t h p nh ng gì b n s h c trong ch ng này
b n s có th t o ra m t s l a ch n màu n gi n Và m t l p cùstom button cái ó
có th

c s d ng l i t d án này n d án khác.

The Graphics Class

(l p h a)

B n có th s d ng l p này xác nh các i m, các ki u tô màu, các ki u ng
v , v vòng tròn hay m t hình dáng nào ó, t ng t nh khi b n s d ng các công
c v tô c a flash.

Tr c khi b t u, chúng ta nên nghe m t l i khuyên, ó là b n ang thao tác v i mã
l nh b ng cách vi t AS, chúng ta t o ra m t hình d ng nh t nh trên m t không gian
xác nh (trong MovieClip ho c trên stage...) Tuy nhiên v trên stage (sách nó vi t là
timeline chính) là m t i u không nên, chúng ta nên t o ra 1 MovieClip (ho c sprite)
và v cho MC ó, chúng ta có th có nhi u tính linh ho t và s c m nh n a khi nó i
n trình bày nh ng thao tác và nh ng hi u ng c a movieclip

T t c các ph ng pháp c a l p graphics này b n bu c ph i g i n nó tr c khi s
d ng mã l nh:

var sp:Sprite = new Sprite();

var g:Graphics = sp.graphics;

ây i t ng g c hi u là m t i t ng graphics c a flash, b n có th thao tác
mã l nh v i nó m t cách c l p. i t ng sp là m t sprite, chúng ta t o ra sau
này g n h a c a g cho nó.

Chúng ta s d ng các ph ng pháp c a l p graphics b ng cách:

g.<method>;

BIÊN SO N L IT SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE

N u khôn mu n thông qua i t ng g b n có th h a cho sprite **descriptiveSpriteName** (tên sprite c miêu t) b ng cách thêm các câu l nh sau:

```
var descriptiveSpriteName:Sprite = new Sprite();
descriptiveSpriteName.graphics.<method>;
descriptiveSpriteName.graphics.<method>;
//l p l i các câu l nh g i l p graphics
descriptiveSpriteName.graphics.<method>;
descriptiveSpriteName.graphics.<method>;
M t cách ng n g n chúng ta dùng thêm l nh with cho descriptiveSpriteName
var descriptiveSpriteName:Sprite() = new Sprite();
with (descriptiveSpriteName.graphics) {
    <method>;
    <method>;
    //l p l i các câu l nh g i l p graphics
    <method>;
    <method>;
}
```

<method> là các ph ng pháp c a l p, ph ng pháp ji thì chúng ta s th o lu n ngay bây gi :

Drawing Lines

(v ng th ng)

ây là thao tác c b n nh t, vi c chúng ta c n làm là xác nh 2 i m u mút c a m t o n th ng, sau ón i 2 i m ól i b ng m t ng th ng v i màu, r ng, ki u nét n i, m c (alpha)... là tùy chúng ta x lý, mã l nh:

```
var sp:Sprite = new Sprite();
addChild(sp);
var g:Graphics = sp.graphics;
g.lineStyle(2, 0x000000);
```

chúng ta có câu l nh th 4 là m i, nó cho chúng ta bi t, r ng c a ng v là 2, màu v tuân theo quy t c c a màu RGB , **0x000000** có ngh a là màu en (c ng là m c nh n u khôn c c p), tham s th 3 là m c (100% là m c nh). Chúng ta có 3 cách xóa ng v a v ó, ho c s d ng ph ng pháp clear ho c cho r ng = 0 ho c ch nh màu v trùng v i màu n n n u ch có n n tr ng (ngoái nét v , trên n n ch a có b t c i t ng nào).

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE J
Bây giờ ta ng chúng ta có 1 cây viết vô hình, và m t ng ta i m có t a (150, 100) n i m có t a (400, 100);, u tiên chúng ta ch m cây vi t t a (150, 100);

g.moveTo(150, 100);

sau ó rẽ cây vi t n i m c n n:

g.lineTo(400, 100);

th là xong, nét v có r ng là 2 và có màu en.

Thêm m t vài VD:

g.lineTo(400, 120);

g.lineTo(150, 120);

g.lineStyle(4, 0xFF0000);

g.moveTo(150, 175);

g.lineTo(400, 175);

dòng 1 nh th 3 c a mă 1 nh thêm vào trên nh m t ng r ng nét v và màu nét v , nó thi t l p các thu c tính này thay cho dòng 1 nh th 4 mă 1 nh ban u.

Chúng ta phân bi t 2 ph ng pháp sau:

moveTo: a cây bút vô hình c a chúng ta t i m t i m, i m ó coi nh là m t d u ch m r t nh trên màn hình mà chúng ta không nhìn th y, ch bi t t a c a nó thoai.

lineTo: V ng th ng n i m có t a xác nh (b ng tham s c a **lineTo**) t m t i m mà chúng ta ã **moveTo** n tr c ó. Sau khi **lineTo** T a cây bút c a chúng ta thay i khôn g còn là t a c a ph ng pháp **moveTo** mà là t a m i trong ph ng pháp **lineTo**.

Drawing Curves

(v ng cong)

Trong trí t ng t ng c a b n, v khôn g ch gi i h n là n m ng th ng mà còn hi u thêm v cách v m t ng cong

M t ng cong có d ng nh sau:

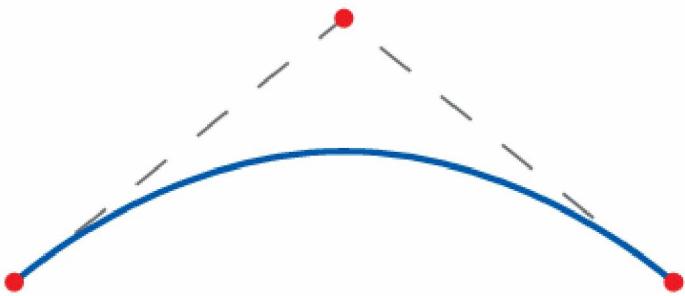


Figure 8-1. A quadratic Bézier curve with one control point for both end points of a line segment

```
g.lineStyle(2, 0x0000FF);
```

```
g.moveTo(150, 100);
```

```
g.curveTo(275, 0, 400, 100);
```

```
g.moveTo(0, 0);
```

câu lệnh `g.moveTo(0, 0);` ch là m t bi n pháp ng n ch n nh ng mă l nh sau có th làm thay i VD mà chúng ta ang xét b ng cách a t a ang v r i kh i i m mút ng cong và a nó v t a (0;0), i u này xem ra “th a” nh ng no s có tác d ng t t trong nhi u tr ng h p khác.

i m u tiên chúng ta ang ch m là (150;100) (trên hình là i m màu bên trái) Chúng ta v m t ng tròn m t cách t ng i (ngh a là không ph i t t c m i

ng tròn u “tròn” theo ý chúng ta mu n) hàm `curveTo` ch cho chúng ta thêm t a c a m t i m, g i nó là tiêu i m c ng c (trên hình minh h a là cái i m n m trên cùng – th hi n b ng 2 tham s u tiên), t ó k hai ng th ng (g ch t) và ng cong c a chúng ta s men theo 2 ng g ch t ó và t ng u n thành ng cong n i m mút th 2 c a nó(i m màu bên ph i – t a c th hi n 2 tham s sau cùng c a hàm), tóm l i chúng ta có 3 i m c n ph i thi t l p cho vi c v m t ng cong n gi n.

Adding Solid Fills

(Tô màu)

Chúng ta báo cho flash bi t khi nào ph i tô màu b ng câu lệnh:

```
beginFill(mă màu-RGB,1);
```

trong ó mă m u là m t s nguyên vi t d i d ng c s 16 tuân theo quy t c màu c a h th ng màu RGB, tham s th 2 là alpha t c là mă c c a màu tô, m c nh là 1 (100% tô y th hi n rõ nét J)

chúng ta báo cho flash bi t vi c tô màu k t thúc b ng 1 nh:

```
endFill()
```

Xét VD sau:

B N D C H NÀY C VI T B I HUYETSATHIENHA – thanh_vinh648@yahoo.com
12/1 – THPT QU C H C HU - B N D C H L U HÀNH TRÊN <http://WWW.VNFX.COM>

var triangle:Sprite = new Sprite();

with (triangle.graphics) {

lineStyle(0);

beginFill(0xFF9900,1);

moveTo(50, 0);

lineTo(100, 100);

lineTo(0, 100);

lineTo(50, 0);

endFill();

}

triangle.x = 50;

triangle.y = 250;

addChild(triangle);

a vào flash ch y th ch c các b n ā hi u mā l nh này có t/d gì roài.

Chúng ta th y c ti n ích c a vi c g n ò h a cho m t i t ng c th ó

Drawing Shapes

(V nh ng hình d ng)

th c hi n vi c v các hình d ng n gi n (tròn, vuông...) chúng ta ph i làm nh các ph n tr c thì r t m t th i gian, flash cho phép chúng ta v nhanh các hình ó b ng các ph ng pháp sau:

drawCircle(X,Y,R): v m t hình tròn t a x,y và có bán kính là R

drawRect(X,Y,A,B): v hình ch nh t t a x,y và có chi u dài A, chi u r ng B

drawRoundRect(X,Y,A,B,C): V hình ch nh t nh ng c nh b làm cong i, cong bao nhiêu thì tùy vào tham s C, C = 0 => hình ch nh t, C = (ho c m t giá thích h p) chúng ta có hình thoí.

M t s hình v b ng h th ng v API:

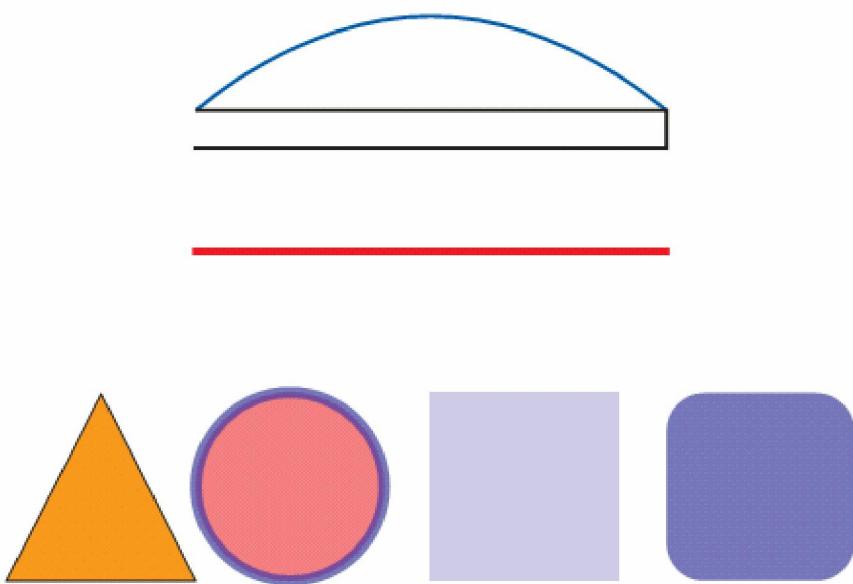


Figure 8-2. The culmination of several Graphics class method calls

Using Gradient Fills

(s d ng ch tô màu hòa tr n)

s d ng ch c n ng này, chúng ta có ph ng pháp:

beginGradientFill()

Nó có 4 tham s , chúng ta s nói n t ng cái 1,

u tiên là gradType, t c là ki u tô, chúng ta có các ki u quyen thu c nh RADIAL (tô màu ki u t a tròn) LINEAR (ki u ng th ng)

Ti p là colors, t c là màu g m có 2 màu, màu ban d u và màu s chuy n b ng cách tô gradient.

Ti p là alphas, là m c, c ng có 2 ph n ng alpha c a màu g c và màu s chuy n.

cu i cùng là ratios, ph n u ch v trí s b t u chuy n màu, ph n cu i ch v trí s k t thúc chuy n màu

ti n chúng ta s các tham s colors, alphas, ratios d i d ng m ng (có th chia làm 2 ph n t)

Chúng ta xét VD sau:

```
var gradType:String = GradientType.RADIAL;  
var colors:Array = [0xFF0000, 0x000000];  
var alphas:Array = [1, 1];  
var ratios:Array = [0, 255];  
var canvas = new Sprite();
```

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE

```
canvas.graphics.beginGradientFill(gradType, colors, alphas, ratios)
canvas.graphics.drawRect(0, 0, 100, 100);
canvas.x = canvas.y = 100;
addChild(canvas);
```

Nh ng gì b n ā h c khōng ph i là t t c , chung ta ch a th i u khi n màu chuy n úng theo v trí mong mu n. Nh ng gì b n c n là l p ma tr n toán h c và gói hình h c (Geometry package) chung ta s th o lu n ti p ph n sau.

Simulating the Pencil Tool

(óng vai là m t cây vi t chì :D)

Ph n này r t thú v chung ta s s d ng nh ng gì ā h c làm m t d án n gi n: Chung ta v nh ng ng th ng theo chu t khi chu t ang nh n, chung ta v nê tr c ti p màn hình chính ch khōng g n h a vào m t i t ng c th nào c , i u ó bu c chung ta g n h a cho i t ng this

```
var drawing:Boolean = false;
```

```
this.graphics.lineStyle(1, 0x000000);
```

```
this.graphics.moveTo(mouseX, mouseY);
```

```
this.addEventListener(Event.ENTER_FRAME, onLoop, false, 0, true);
stage.addEventListener(MouseEvent.MOUSE_DOWN, onDown, false, 0, true);
stage.addEventListener(MouseEvent.MOUSE_UP, onUp, false, 0, true);
```

```
function onDown(evt:MouseEvent):void {
```

//khi chu t è xu ng, nh ng nét v s hi n ra n t a c a chu t
drawing = true;

```
}
```

```
function onUp(evt:MouseEvent):void {
```

//khi chu t è xu ng, khōng có nét v n n ào c , thay vào ó t a i m v c xác
nh //theo chu t
drawing = false;

```
}
```

```
function onLoop(evt:Event):void {
```

```
if (drawing) {
```

```
    this.graphics.lineTo(mouseX, mouseY);
```

```
    } else {
        this.graphics.moveTo(mouseX, mouseY);
    }
}
```

Th y thú v y ch K.

The Geometry Package

(gói hình h c)

T c là gói **flash.geom** là m t gói cho phép chúng ta t o ra các i m, hình ch nh t ... hay s bi n i th hi n c a i t ng. Chúng ta quan tâm n các l p sau c a gói: **Points, Rectangles, Matrices**. Gói hình h c s c nói thêm ch ng sai th o lu n trong ph n màu s c (color).

Creating Points

(t o m t i m)

M t i m bao g m t a x và y c a nó

Chúng ta t o ra m t i m theo ki u m ng b ng cách sau:

```
var arrayPoint:Array = new Array(0, 0);
```

M ng có 2 ph n t t ng ng t a x và y c a i m ó.

```
trace(arrayPoint[0], arrayPoint[1]);
```

Chúng ta có c a s output giá tr 0 0 vì m ng ây có 2 ph n t mang giá tr 0

Ngoài ra chúng ta có nh ng cách khác kh i “phi n” n array, 2 VD sau:

```
var objPoint:Object = {x:0, y:0};
```

```
trace(objPoint.x, objPoint.y);
```

```
var objPoint2:Object = new Object();
```

```
objPoint2.x = 0;
```

```
objPoint2.y = 0;
```

```
trace(objPoint2.x, objPoint2.y);
```

2 i m c khai báo trên theo ki u Object, t t nhiên nó có nhi u thu c tính khác nhau và chúng ta khai báo thêm cho nó là x và y. B n hâ y ý câu l nh trace, chúng ta có c a s output 2 kq gi ng nhau:

```
//0 0
```

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE J
là ki u qu c a **objPoint** và **objPoint2** vì x và y ch n thu n là 2 thu c tính n m
trong i t ng ch a 2 thu c tính (**objPoint**) hay nó là m t thu c tính c a i t ng
hoàn ch nh (**objPoint2**)

Dùng làm cách nào i n a thì khi truy su t n l i m trên flash, chúng ta v n c n ít
nh t 2 giá tr ch x và y, tuy nhiên v i khai báo ki u i t ng **Point** Chúng ta s
ch c n l bi n l u giá tr x và y, giá tr m c nh cho t a 1 i m m i c khai
báo là 0 0

```
var pt:Point = new Point();
```

```
trace(pt.x, pt.y);
```

```
//0 0
```

Chúng ta th y r ng pt v n có 2 thu c tính c a nó ch giá tr x và y

Xét VD sau:

```
var pt2:Point = new Point(100, 100);
```

```
trace(pt2);
```

```
//(x=100, y=100)
```

Giá tr pt2 mang trong nó 2 giá tr x và y

Chúng ta s làm ti p VD sau:

```
var pt1:Point = new Point(100, 100);
```

```
var pt2:Point = new Point(400, 400);
```

```
pt1.offset(50, 50);
```

```
trace(pt1);
```

```
//ch c n ng offset cho chúng ta t ng thêm giá tr c a x và y lên theo n v pixel
```

```
//(x=150, y=150)
```

```
trace(pt1.add(pt2));
```

```
//ch c n ng add cho phép thêm giá tr c a x và y b ng cách c ng vào giá tr c a i m cho  
tr c
```

```
//(x=550, y=550)
```

```
trace(pt2.subtract(pt1));
```

```
//ch c n ng subtract cho phép gi m giá tr c a x và y b ng cách tr giá tr c a i m cho tr c
```

```
//(x=250, y=250)
```

```
trace(pt1.equals(pt2));
```

```
//h i xem 2 i m pt1 và pt2 có trùng nhau hay không (true n u 2 i m trùng nhau)
```

```
// i u này n gi n hóa khi ph i if cho t a x và y
```

```
//false
```

```
trace(Point.distance(pt1, pt2));
```

```
//tính kho ng cách
```

trace(Point.interpolate(pt1, pt2, .5));

//xác nh m t i m n m trên o n th ng có 2 u là pt1 và pt2 (.5=> gi a o n //th ng, 1 thì i m ó có t a = pt1, 0 thì i m ó có t a = pt2... cái tham s th //3 này là tùy mìn xác nh i m mong mu n)

//(x=275, y=275)

i m v n xác nh khi chúng ta t o hay gán các giá tr cho các thu c tính c a nó nh x hay y (vd **pt1.x**), i u này giúp l p trình viên không g p khó kh n khi s d ng

i m hay s d ng thu c tính c a i t ng

i u này r t ti n l i cho vi c t o m t i m trên màn hình, chúng ta có th s d ng chúng không gi i h n cho các i t ng. Chúng ta s h c thêm ch ng sau các k thu t làm vi c v i bitmap, chúng ta s th y c s a d ng c a k thu t này.

Creating Rectangles

(t o hình ch nh t)

T ng t v t o m t i m m t hình ch nh t trong flash ch a 4 tham s chính: x,y,w (chi u dài) ,h (chi u r ng).

var rect:Rectangle = new Rectangle(0, 0, 100, 100);

trace(rect.x, rect.y);

//0 0

//s d ng thu c tính

trace(rect);

//(x=0, y=0, w=100, h=100)

//s d ng cách t o hình ch nh t c a gói hình h c

Chúng ta có các hàm c b n v i m t i t ng Rectangle:

var rect:Rectangle = new Rectangle(50, 50, 200, 100);

trace(rect.left, rect.top, rect.right, rect.bottom);

//kho ng cách t c nh // Oy bên trái n tr c Oy , kho ng cách t c nh // Ox bên trên // n tr c Ox, kho ng cách t c nh // Oy bên ph i n tr c Oy, kho ng cách t c nh // //Ox bên d i n tr c Ox

//50 50 250 150

trace(rect.topLeft, rect.bottomRight);

//t a c a nh trái phía trên và nh ph i phía d i

//(x=50, y=50) (x=250, y=150)

trace(rect.x, rect.y, rect.width, rect.height);

//50 50 200 100

//các thông s c b n c a hình ch nh t

BIÊN SO NL IT SÁCH LEARNING ACTION SCRIPT BEGINNER ‘S GUIDE
Chúng ta có thể sử dụng phép offset để tính giá trị cho x và y thông qua m