قواعد کد نویسی در فلش (اکشن اسکریپت)

Code Style
in FLASH (ActionScript)

گردآوري، تاليف و ترجمه:

امين كريمي

www.amindesign.ir

صاحب امتياز:

www.amindesign.ir

۱۳۸۶

الف) قواعد نام گذاري

۱) متغيرها:

- Actionscript به بزرگي و كوچكي حروف حساس است بطور مثال myBall و MyBall رو متغير متفاوت هستند ولي در عين حال دقت كنيد براي جلوگيري از گيج شدن برنامه نويس براي متغيرهاي خود نا م يكسان با اندازه حروف متفاوت انتخاب نكنيد.

- كلمات را درست اختصار كنيد كه مفهوم چند چيز را نرساند، به عنوان مثال sec هم نماد section است و هم نماد second.

- از كوته سازي نامها پرهيز شود.

مثال:

computeAverage(); *// NOT: compAvg();*

واژه هاي رايج هيچگاه نبايد بصورت كوتاه شده نوشته شوند. كوته‌سازي‌هاي زير غيرمجاز مي باشند:

cmp بجاي command

cp بجاي copy

pt بجاي point

comp بجاي compute

init بجاي initialize

همچنين كوته نامها يا سرنامهاي معروف نبايد به صورت كامل نوشته شوند. مثلاً هرگز نبايد:

HypertextMarkupLanguage را بجاي html،

CentralProcessingUnit را بجاي cpu و يا

PriceEarningRatio را بجاي pe نوشت.

- نام متغير را با حرف كوچک آغاز كنيد.

- براي نام هاي چندكلمه اي، كلمات را به هم چسبيده بنويسيد بطوريكه حرف اول تمام كلمات بجز كلمه اول بزرگ باشد.

myFirstVar

- نام متغيرها تنها مي تواند شامل حروف، اعداد و علامت دلار ($) باشد.

- نام متغير را با عدد آغاز نكنيد.

مثال غلط:

my var , my/var , 5 var , 25var

مثال درست:

myVar, var25

- كلماتي را كه جزئي از زبان اكشن اسكريپت هستند (keywords) را براي نام متغير انتخاب نكنيد.
  همينطور در مورد كلمات مربوط به ساختارها و اجزاي فلش

مثال غلط:

Var = "2"

Return = "flash"

Interface = 10

TextField = "my var"

- از كدهاي رزوز شده بپرهيزيد!

add	and	break	case
catch	class	continue	default
delete	do	dynamic	else
eq	extends	false	finally
for	function	ge	get
gt	if	ifFrameLoaded	implements
import	in	instanceof	interface
intrinsic	le	it	ne
new	not	null	on
onClipEvent	or	private	public
return	set	static	super
switch	tellTarget	this	throw
try	typeof	var	void
while	with		

as	abstract	Boolean	bytes
char	const	debugger	double
enum	export	final	float
goto	is	long	namespace
native	package	protected	short
synchronized	throws	transient	use
volatile			

- در اکشن اسکریپت نیازي به گذاشتن کلمه var قبل ازنام متغیر و تعیین نوع آن بعد ازآن نیست ولي ترجیحا براي خوانا تر بودن و سریع تر اجرا شدن کدها این کار را بکنید.

```
var font_array:Array = TextField.getFontList()
```

۲) مقادیر ثابت:

- براي نام گذاري مقادير ثابت از حروف بزرگ استفاده كنيد.

- در نامهاي چندكلمه اي مقادير ثابت (كه تماما بايد با حروف بزرگ نوشته شوند) كلمات را با under score (_) از هم جدا كنيد.

BASE_NUMBER = 10

۳) توابع:

- نام توابع بايد با حروف كوچک آغاز شود.

- در صورت چند كلمه اي بودن حرف اول تمام كلمات آن غير از كلمه اول بزرگ نوشته مي شود.

gerCurrendSound()

- نام گذاري توابع و متغير ها ظاهرا يكسان مي باشد با اين تفاوت كه در پايان نام توابع () قرار مي گيرد.

- براي توابعي كه يک مقدار را محاسبه مي كنند مي توان از واژه compute بهره جست.

مثال:

```
valueSet->computeAverage();
matrix->computeInverse();
```

- براي توابعي كه يک مقدار را جستجو مي كنند مي توان از واژه find بهره جست.

مثال:

```
vertex.findNearestVertex();
matrix.findMinElement();
```

- براي توابعي كه براي مقداردهي اوليه استفاده مي شوند، مي توان از واژه initialize بهره گرفت.

مثال:

```
printer.initializeFontSet();
```

نبايد از گونه كوتاه شدهي "init" بجاي "initialize" استفاده كرد.

۴) متدها:

- نام گذاري متدها همانند توابع است كه بايد شامل فعل متد باشد.

Sing()
runFast()

- براي متدهايي كه عملياتهاي مكمل يا ضد يكديگر را انجام مي دهند بايد نامهاي متقارن و مناسب انتخاب نمود.

مثال:

get/set, add/remove, create/destroy, start/stop, insert/delete, increment/decrement, old/new, begin/end, first/last, up/down, min/max, next/previous, old/new, open/close, show/hide, suspend/resume

۵) حلقه ها:

- براي افزايش سرعت و حجم كم ساختار حلقه، براي متغير هاي دستور حلقه از نام هاي يک حرفي (i, j, k, m, n براي integer .و c ,d براي كاراكتر) استفاده كنيد.

```
for (int i = 0; i < nTables); i++){
    …. }
```

۶) کلاس ها و اشیا:

- نام کلاس ها را با حروف بزرگ آغاز کنید.
- در نام هاي چند کلمه اي حرف اول تمام کلمات بزرگ باشد.
- نام کلاس باید ساده و توصیفي از عملکرد کلاس باشد.

```
class Birds
class StreamingVideo
```

- نام کلاس نباید زیرمجموعه اي از ویژگي هاي آن باشد.
- قبل از نام متغیرهاي داخل کلاس از پیشوند _m استفاده کنید.
- هنگامي که قصد دستیابي به یکي از متغیرها یا مشخصه‌هاي کلاس را داشته باشیم باید از عبارات get/set استفاده شود.

مثال:

```
employee.getName();
employee.setName(name);
matrix.getElement(2, 4);
matrix.setElement(2, 4, value);
```

۷) بسته ها (Packages):

- نام بسته ها را با mx يا com.macromedia آغاز كنيد. قسمت بعدي شامل نام رويه است. براي مثال:

  mx.containers.ScrollPan

- اگر بسته، ساخته خود شماست نام آن را با com.macromedia آغاز كنيد:

  com.macromedia.Shapes

۸) كامپوننت ها (Components):

- نام كامپوننت ها با حروف بزرگ آغاز مي شود.
- در نام هاي چندكلمه اي حرف اول تمام كلمات بزرگ نوشته مي شود.

CheckBox
DataGrid
DataField

۹) Interfaces

- نام Inteface ها با حروف بزرگ آغاز مي شود.
- در نام هاي چندكلمه اي حرف اول تمام كلمات بزرگ نوشته مي شود.
- نام Interface ها معمولا بصورت صفت بيان مي شود.

Interface EmployeeRecords

۱۰) قواعد عمومي

- نامها بايد به زبان انگليسي باشد. از نوشتن به "PINGLISH" يا زبانهاي ديگر پرهيز شود.

- براي متغيرهايي ک محدوده گستردهاي دارند بايد نام بلند انتخاب شود.

- نامهايي که بيانگر مجموعهاي از اشياء مي باشند بايد بصورت جمع ظاهر شوند.

- براي نام هايي که نشاندهنده تعداد مي باشند بايد از پيشوند n استفاده نمود.

مثال:

nPoints, nLines

- براي متغيرهاي نشاندهنده عدد (شماره يک شيء) از پسوند No استفاده شود.

مثال:

tableNo, employeeNo

يک جايگزين براي اين روش استفاده پيشوند i مي باشد؛ مانند: iTable و iEmployee . اين روش بخصوص در حلقه ها و براي متغيرهاي شمارنده مفيد است.

- براي نام متغيرهاي boolean بايد از پيشوندهاي مناسب مانند: is، has، can، should و ... استفاده نمود.

مثال:

isSet, isVisible, isFinished, isFound, isOpen, hasLicense, canEvaluate, shouldSort

- از نام گذاري متغيرهاي boolean بصورت منفي خودداري شود.

مثال:

isError; *// NOT: isNoError*

isFound; *// NOT: isNotFound*

نام گذاريهاي منفي در هنگام اعمال عملگر منطقي "!"، باعث کاهش خوانايي کد مي گردند.

ب) Include

- در برنامه نويسي با فلش (اكشن اسكريپت) مي توان كدهاي خارجي را با اشتراک گذاشتي اين كار با دستور include# و به اين صورت انجام مي شود:
#include "codes.as"

- فايل كد خود را بايد با پسوند as. ذخيره سازي كنيد.
- تمام فايل هاي مطلوب را در ابتداي كد اصلي و در خطوط پشت سر هم include كنيد.

پ) انواع (Types)

- در يک کلاس ابتدا public سپس private مي آيد.
- براي تعريف يک متد يا متغير به شکل public يا private اين کلمات را قبل از آن قرار دهيد.

```
public function userClass(username,password){
:
}
private var m_username,m_password
```

- تبديل انواع بايد بصورت صريح انجام شود (نه ضمني)

ت) متغير ها

- متغيرها به هنگام اعلان بايد مقدار دهي اوليه شوند. با اين كار اطمينان حاصل مي شود كه متغيرها همواره معتبرند. گاهي اوقات مقدار دهي اوليه متغيرها در هنگام اعلان غير ممكن مي باشد كه در اين صورت بهتر است دستنخورده باقي بماند.

- متغيرها نبايد به گونه اي باشند كه چند معنا از آنها برداشت شود.

اين كار باعث مي شود تا همه مفاهيم بصورت يكتا معرفي شده باشند كه خود سبب خوانايي كد مي گردد.

- استفاده از متغيرها، توابع سرتاسري (global) بايد به كمترين حد خود برسد.

- متغيرهاي خاص كلاس نبايد به صورت public معرفي شوند.

- متغيرها بايد در كوتاهترين دامنه ديد تعريف گردند .

اين كار باعث مي شود كه تأثيرات جانبي متغير در طول كد بهتر و ساده تر كنترل گردد.

- براي اينكه صفر بودن يا نبودن يک متغير را بسنجيم نبايد از صورت ضمني عبارت شرطي استفاده كرد. مگر در مواقعي كه متغير ازنوع boolean باشد.

مثال:

```
if (nLines != 0) // NOT: if (nLines)
if (value != 0.0) // NOT: if (value)
```

ث) كدگذاري روي سمبل ها و فريم ها

- كدهاي برنامه هاي ساده را حتي الامكان روي سمبل ها بنويسيد.
- در كد نوشتن روي button ها يا movie clip ها به روابط پدر-فرزندي و آدرس دهي صحيح دقت داشته باشيد.
- در پروژه هاي نسبتا طولاني كل كد خود را در يک فريم و روي خط زمان اصلي بنويسيد. اين كار سبب خوانائي بيشتر برنامه و تغيير ساده آن مي شود.
- كد گذاري روي يک سمبل با اين ساختار انجام مي شود.

```
on (press) {
:
}
```

- كدگذاري يک سمبل روي فريم با اين صاختار انجام مي شود.

```
M1.onPress = function (){
:
}
```

ج) حلقه ها

- تنها متغيرهاي حلقه مي توانند در داخل دستور for() استفاده گردند.

مثال:

```
sum = 0;                // NOT: for (i = 0, sum = 0; i < 100; i++)
for (i = 0; i < 100; i++)    //     sum += value[i];
  sum += value[i];
```

اين كار سبب افزايش خوانايي و سادگي نگهداري و گسترش كد مي گردد.

- متغيرهاي حلقه بايد دقيقاً قبل از حلقه مشخص شوند

مثال:

```
isDone = false;          // NOT: bool isDone = false;
while (!isDone) {        //  :
  :                      // while (!isDone) {
}                        //  :
                         // }
```

- مي توان از حلقه هاي do-while استفاده نكرد.

به سبب قرار داشتن شرط حلقه‌هاي do-while در انتهاي آنها، خوانايي اين حلقه ها از حلقه هاي while و for كمتر است . همچنين مي توان هر حلقه do-while را به حلقه‌هاي while يا for تبديل نمود. استفاده از تعداد ساختارهاي كمتردر داخل برنامه سبب افزايش خوانايي آن مي گردد.

- از استفاده از break و continue در داخل حلقه ها پرهيز شود.

تنها در صورتي استفاده از break و continue مجاز است كه ثابت شود بكار بردن آنها باعث افزايش خوانايي برنامه خواهد شد.

- براي نشان دادن حلقه هاي بي پايان از حلقه while(true) استفاده شود.

مثال:

```
while (true) {
:
}
```

```
for (;;) { // NO!
:
}

while (1) { // NO!
:
}
```

چ) دستورها و عبارات شرطي

- از استفاده از عبارات شرطي در هم و پيچيده خودداري شود در عوض مي توان از متغيرهاي موقتي و كمكي boolean ياري گرفت. با اين كار كد برنامه، خود- مستند مي گردد و همچنين خواندن، اشكالزدايي و نگهداري كد آسانتر مي گردد.

- در دستورهاي if-else ، بخشي كه انتظار اتفاق افتادنش وجود دارد (بخش مثبت) در قسمت if و بخش استثناء (منفي) شرط در قسمت else آورده مي شود.

مثال:

```
if (isOk) {
..
}
else {
:
}
```

- دستورات شرطي بايد در خطوط مجزا نگاشته شوند.

مثال:

```
if (isDone)      // NOT: if (isDone) doCleanup();
   doCleanup();
```

اين كار در هنگام اشكال زدايي مفيد واقع مي شود چرا كه اگر مقدم و تالي در يک خط واقع شده باشند، نمي توان تشخيص داد كه شرط درست بوده يا نادرست.

- از آوردن دستورات و عبارات اجرايي در دستورات شرطي بايد خودداري شود.

خواندن دستورات شرطي شامل دستورات اجرايي مشكل مي باشد؛ بويژه براي تازه كاران.

ح) موارد ديگر:

- در برنامه نويسي با فلش نيازي به گذاشتن سمي كالن (;) در پايان دستورات نمي باشد و نوشتن هر دستور در خط مجزا كفايت مي كند ولي براي رعايت استاندارد و خوانائي بيشتر از آن استفاده كنيد.

- پنل Action فلش گزينه اي به نام Auto format وجود دارد. مي توانيد با زدن اين دكمه كد خود را بصورت استاندارد درآوريد. براي درست كردن استاندارد مطلوب خود از منوي pereferences در پنل اكشن تب مربوط به Auto format را انتخاب كرده و style موردنظر خود را بسازيد.

- استفاده صريح از اعداد و ارقام – به غير از ارقام 0 و 1 -- در داخل كد مجاز نمي باشد و بجاي آنها بايد از ثابتهايي با نام معنادار بهره گرفت.

اگر عددي به خودي خود معنايي نداشته باشد، استفاده از ثابتي با نام با معنادار مي تواند سبب خوانايي كد گردد.

- اعداد از نوع اعشاري بايد شامل حداقل يک رقم صحيح و ممیز (نقطه اعشاري) باشند.

با اين روش متغيرهاي از نوع اعشاري از متغيرهاي صحيح بازشناخته مي شوند.

- در اعداد اعشاري حتماً بايد دست كم يک رقم صحيح پيش از ممیز وجود داشته باشد.

مثال:

```
double total = 0.5; // NOT: double total = .5;
```

خ) بلاک ها

- مقدار پايه فرو رفتگي برابر دو ستون مي باشد.

مثال:

```
for (i = 0; i < nElements; i++)
  a[i] = 0;
```

مقدار كمتر از دوستون سبب ناتواني در شناسايي بلاكها مي شود. همچنين مقداربيشتر از چهارستون نيز باعث دشواري خواندن و همچنين افزايش احتمال گسسته شدن خطوط در بلاكهاي تودرتو مي گردد.
بهترين مقادير براي انتخاب دو، سه يا چهار ستون مي باشد.

- نحوه نوشتن بلاكها بايد مطابق دو نمونه زير باشد.(كه نمونه اولي بهتر مي باشد)

```
while (!done) {
  doSomething();
  done = moreToDo();
}
```

يا

```
while (!done) {
  doSomething();
  done = moreToDo();
}
```

همچنين يک بلاک نبايد به گونه زير نوشته شود:

```
while (!done)
  {
    doSomething();
    done = moreToDo();
  }
```

- نحوه نوشتن بلاک كلاس بايد به شكل زير باشد:

```
class SomeClass {
:
```

```
}
```

- نحوه نوشتن بلاک توابع باید به شکل زیر باشد:

```
void someMethod(){
:
}
```

- نحوه نوشتن بلاک دستور if-else باید به شکل زیر باشد:

```
if (condition) {
  statements;
}

if (condition) {
  statements;
}else {
  statements;
}

if (condition) {
  statements;
}else if (condition) {
  statements;
}else {
  statements;
}
```

- نحوه نوشتن بلاک دستور for باید به شکل زیر باشد:

```
for (initialization; condition; update) {
  statements;
}
```

- نحوه نوشتن دستور for تهي، بايد به شكل زير باشد:

```
for (initialization; condition; update)
 ;
```

- نحوه نوشتن بلاک دستور while بايد به شكل زير باشد:

```
while (condition) {
 statements;
}
```

- نحوه نوشتن بلاک دستور do-while بايد به شكل زير باشد:

```
do {
 statements;
} while (condition);
```

- نحوه نوشتن بلاک دستور switch بايد به شكل زير باشد:

```
switch (condition) {
 case ABC :
  statements;
  break;

 case DEF :
  statements;
  break;

 default :
  statements;
  break;
}
```

- ساختارهاي تک دستوري if-else، while يا for مي توانند بدون آكولاد، "{}"، نوشته شوند.

مثال:

```
if (condition)
  statement;

while (condition)
  statement;

for (initialization; condition; update)
  statement;
```

د) فضاهاي خالي

- پيش و پس از عملگرها بايد يک فضاي خالي باشد.
- پس از واژگان پس‌نهاده C++ (reserved words) بايد يک فاصله قرار گيرد.
- پس از هر کاما بايد يک فضاي خالي قرار گيرد.
- پيش و پس از علامت دو نقطه، ":"، بايد يک فاصله قرار گيرد.
- پس از علامت ";" بايد يک فضاي خالي قرار گيرد.

مثال:

```
a = (b + c) * d; // NOT: a=(b+c)*d

while (true)    // NOT: while(true)

doSomething(a, b, c, d);  // NOT: doSomething(a,b,c,d);

case 100 : // NOT: case 100:
for (i = 0; i < 10; i++) { // NOT: for(i=0;i<10;i++){
 ...
```

اين روش با برجسته ساختن مؤلفه هاي منفرد دستورات، سبب خواناتر شدن کد مي گردد.

- اگر پس از نام توابع، يک نام واقع شود، مي توان بعد از نام تابع يک فاصله خالي قرار داد.

مثال:

```
doSomething (currentFile);
```

اين کار با واضح و مشخص ساختن هر نام، خوانايي کد را موجب مي شود. اگر هيچ نامي پس از نام تابع نيامده باشد مي توان فاصله خالي را حذف نمود.

```
doSomething()
```

راه ديگر اين است که فاصله را پس از پرانتز باز قرار داد که بعضي از برنامه نويسان براي وضوح و همچنين تقارن بيشتر، پيش از پرانتز بسته نيز يک فاصله خالي قرار مي دهند.

```
doSomething( currentFile );
```

- درون یک بلاک را باید از نظر منطقي گروه بندي کرد و پس از هر گروه یک خط خالي قرار داد.

مثال:

```
matrix = new Matrix4x4();

cosAngle = Math.cos(angle);
sinAngle = Math.sin(angle);

matrix.setElement(1, 1, cosAngle);
matrix.setElement(1, 2, sinAngle);
matrix.setElement(2, 1, -sinAngle);
matrix.setElement(2, 2, cosAngle);

multiply(matrix);
```

- متغیرها به هنگام اعلان باید چپ- تراز گردند.

- تراز کردن در هر جایي که موجب افزایش خوانایي گردد توصیه مي شود.

مثال:

```
if     (a == lowValue)    compueSomething();
else if (a == mediumValue) computeSomethingElse();
else if (a == highValue)   computeSomethingElseYet();

value = (potential       * oilDensity)   / constant1 +
      (depth             * waterDensity) / constant2 +
      (zCoordinateValue * gasDensity)    / constant3;

minPosition     = computeDistance(min,     x, y, z);
averagePosition = computeDistance(average, x, y, z);

switch (value) {
```

```
  case PHASE_OIL   : strcpy(phase, "Oil");   break;
  case PHASE_WATER : strcpy(phase, "Water"); break;
  case PHASE_GAS   : strcpy(phase, "Gas");   break;
}
```

در بسياري از موارد، فضاهاي خالي مي تواند سبب افزايش خوانايي كد گردد حتي اگر قواعد گفته شده را نقض نمايد. بايد توجه كرد كه در صورت استفاده از فضاهاي خالي بايد ترازمنديها رعايت شود.

ذ) توضيحات (Comments)

- كد بايد خود- مستند باشد.

در حالت كلي كد بايد بقدري خوانا و قابل فهم باشد كه نيازي به استفاده از توضيحات نداشته باشد. با نام گذاريهاي شايسته و رعايت ساختارهاي مناسب اين خواسته تحقق مي يابد.

- تمامي توضيحات بايد به زبان انگليسي نگاشته گردند.

- براي اينكه در ساختار منطقي كد خلل وارد نشود، توضيحات درون بلاک بايد مطابق با تورفتگيها نوشته شوند.

گردآوري و ترجمه:
امين كريمي
www.amindesign.ir