

# Introducción al desarrollo de RIA's con Adobe Flex 3.0 Dia 5

by S. Muñoz-Gutiérrez  
[stalinmunoz@yahoo.com](mailto:stalinmunoz@yahoo.com),  
[informes@grupolinda.org](mailto:informes@grupolinda.org)

Grupo LINDA  
Facultad de Ingeniería

UNAM México Octubre-Diciembre 2009



# Arreglos

Instancias de la clase Array

```
var a:Array = new Array();
```

```
var a:Array = new Array(5); //5 elementos
```

```
var a:Array = new Array(1,3,5,"hola","mundo");
```

```
//elementos pueden ser de tipos diferentes
```

usamos [ ] para acceder a elementos

```
a[3]
```

# Atributos de Arreglos

length indica el tamaño del arreglo

```
var a:Array = new Array();  
a[2] = "hola";  
a[7] = "mundo";  
nums.text = a.length.toString(); // 8
```

# Métodos de Arreglos

**concat** yuxtapone dos arreglos

```
var a:Array = new Array(1,2,3);  
var b:Array = new Array(4,5,6);  
var c = a.concat(b); // 1,2,3,4,5,6  
c = b.concat( 7, 8, 9); // 4,5,6,7,8,9
```

# Métodos de Arreglos

**push** agrega uno o más elementos al final del arreglo

```
var a:Array = new Array(1,2,3);  
var b:Array = new Array(4,5,6);  
a.push(b); // 1,2,3,4,5,6  
a.push(7); // 1,2,3,4,5,6,7  
b.push(7,8); // 4,5,6,7,8
```

# Métodos de Arreglos

**pop** extrae el último elemento de un arreglo

```
var a:Array = new Array(1,2,3);  
var b:int = a.pop(); // 3  
b = a.pop(); // 2  
b = a.pop(); // 1  
b = a.pop(); // 0!
```

# Métodos de Arreglos

**unshift** agrega elementos al inicio del arreglo (no acepta arreglos como argumentos)

```
var a:Array = new Array(1,2,3);  
var b:Array = new Array(4,5,6);  
a.unshift(0); // 0,1,2,3  
a.unshift(-2,-1); // -2,-1,0,1,2,3  
b.unshift(a); // / 0,1,2,3!
```

# Métodos de Arreglos

**shift** extrae el primer elemento de un arreglo

```
var a:Array = new Array(1,2,3);  
var b:int = a.shift(); // 1  
b = a.shift(); // 2  
b = a.shift(); // 3  
b = a.shift(); // 0!
```

# Métodos de Arreglos

**slice(inicio,fin)** extrae un subarreglo del índice inicio al índice fin (**fin no se incluye**)

```
var a:Array = new Array(1,2,3,4,5,6,7);  
var b:Array = a.slice(3,5); // 4,5  
b =a.slice(5); // 6,7
```

# Arreglos Multidimensionales

Elementos del arreglo son arreglos

```
var a:Array = new Array();  
a[0]=[1,2,3,4];  
a[1]=[5,6,7];  
a[2]=[8,9];  
for each(var j:Array in a) {  
    trace(j.toString());  
}
```

```
var b:Array = new Array([1,2,3],[4,5,6],[7,8,9]);  
trace(b.toString());
```

# Sintaxis de ciclos

- **for**
- for in
- for each ... in
- while
- do ... while

Sintaxis tipo C

```
var i: Number;  
for(i=0; i<10; i++){  
    //Instrucciones que  
    //se repiten  
}
```

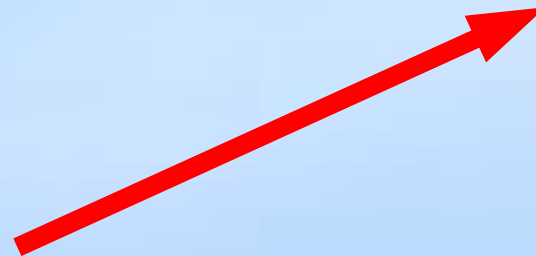
# Sintaxis de ciclos

- for                   Accediendo a propiedades de objetos **genéricos**
  - **for in**
  - for each ... in
  - while
  - do ... while
- ```
var obj:Object =  
    {x:20, y:"hola"};  
for (var i:Object in obj) {  
    trace(i + ": " + obj[i].toString());  
}
```

Extrae los nombre de las propiedades


# Sintaxis de ciclos

- for                   Accediendo a elementos de arreglos
  - **for in**
  - for each ... in
  - while
  - do ... while
- ```
var a:Array = ["hola","mundo!"];  
for (var i:String in a) {  
    nums.text +=(" " +a[i]);  
}
```



Extrae los índices como nombres

# Sintaxis de ciclos

- for
  - for in
  - **for each ... in**
  - while
  - do ... while
- Accediendo a elementos de una colección
- ```
var a:Array = ["hola","mundo!"];  
for each(var i:String in a) {  
    nums.text +=(" " + i);  
}
```
- 

Extrae los elementos de la colección

# Sintaxis de ciclos

- for Se repite 0 o más veces
- for in
- for each ... in
- **while**

```
var i:int = 0;
while (i < 10){
    trace( i++);
}
```
- do ... while

# Sintaxis de ciclos

- for Se repite 1 o más veces
- for in
- for each ... in
- while
- **do ... while**

```
var i:int = 0;  
do {  
    trace( i++);  
} while(i < 10);
```

# Ejercicio

Implemente una interfaz que permita al usuario introducir dos matrices de 3 x 3 y le permita realizar las operaciones de suma, resta y multiplicación. Cada entrada de la matriz debe introducirse con un TextInput utilice un Grid como contenedor de sus componentes. Por ejemplo:

```
<mx:Grid>
```

```
  <mx:GridRow>
```

```
    <mx:GridItem>
```

```
      <mx:TextInput>...
```