

Tehnologii RIA

Cosmin Varlan

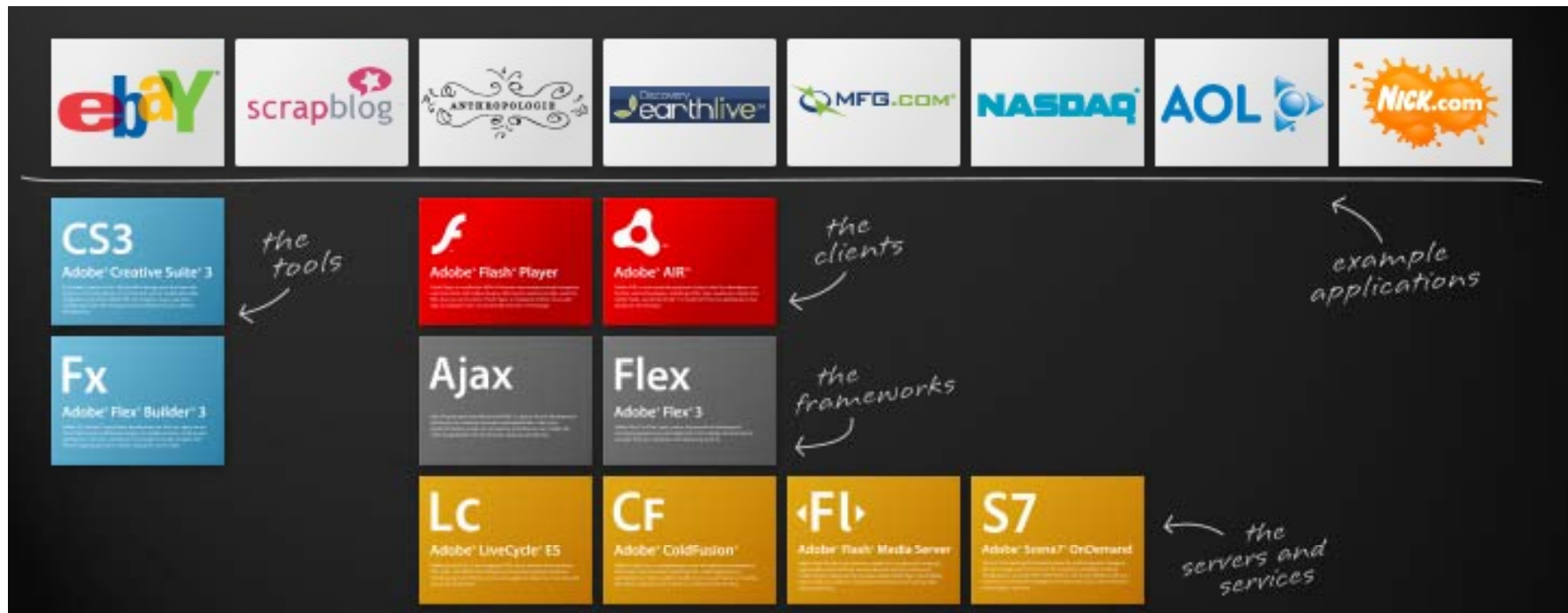
Rich Internet Applications

- Rich Internet applications (RIAs) offer a rich, engaging experience that improves user satisfaction and increases productivity. Using the broad reach of the Internet, RIAs can be deployed across browsers and desktops.
(sursa: www.adobe.com)

Rich Internet Applications - tech

- Java Script / AJAX (+SVG) = RIA
[ex: gmail, noul Yahoo mail]
- Microsoft Silverlight = RIA [chrome?]
http://www.theregister.co.uk/2008/08/18/silverlight_pros_and_cons/
<http://memorabilia.hardrock.com/>
- Adobe Flash, Flex, AIR? = RIA
restul ... (poate exista si altele)

Rich Internet Applications



Rich Internet Applications - Flash



= Adobe Flash CS3

Rich Internet Applications - Flash



- + Crearea de obiecte vectoriale
- + Permite crearea animatiilor in modul de editare timeline
- + Atasare de comportamente obiectelor
- + OOP

Rich Internet Applications - Flash



- + Dimensiuni mici
- + Independenta de platforma
- Sa aiba player flash instalat
- Vectori multi -> procesor incarcat
- Incompatibilitatea cu motoarele de cautare (pentru text static)

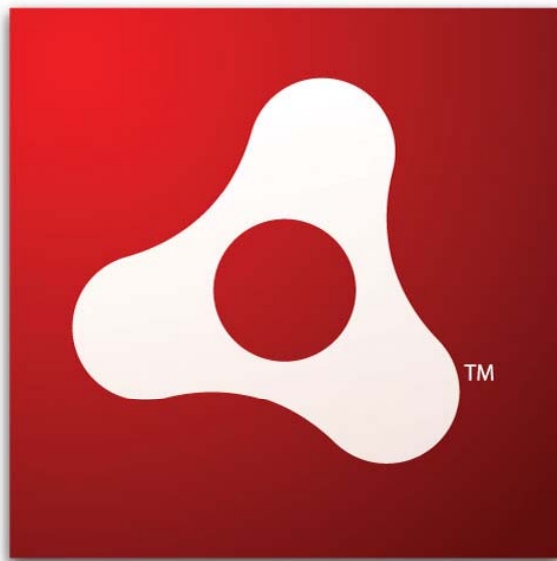
Rich Internet Applications - Flex



- Mediu de programare pentru AS3
- Nu permite editarea grafica a obiectelor (editorul in schimb este “prea tare”)
- Putem realiza proiecte tip Flex / ActionScript
- Bazat pe Eclipse (editor java)

Rich Internet Applications - AIR

- http://www.adobe.com/resources/business/rich_internet_apps/getting_started/

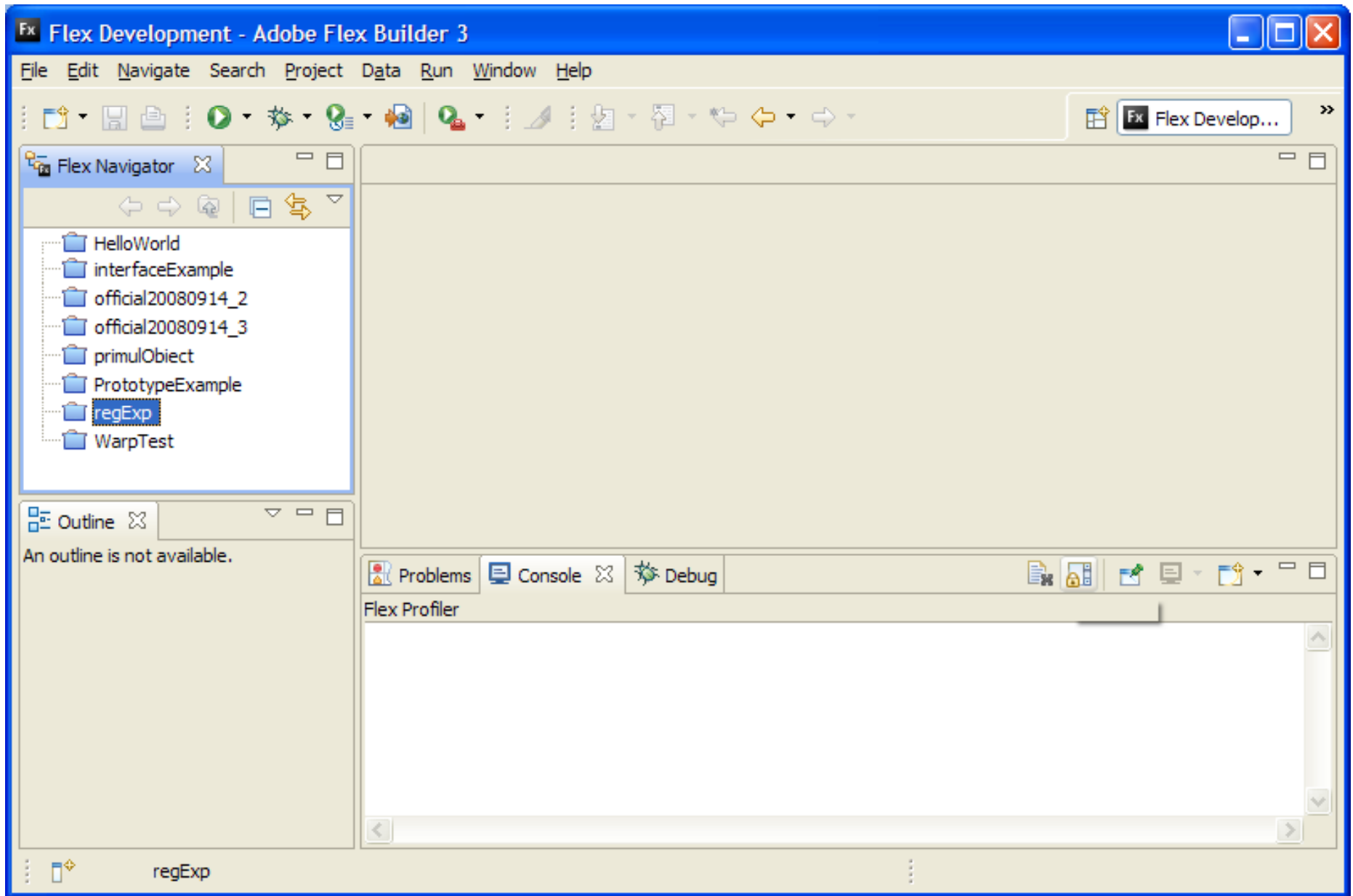


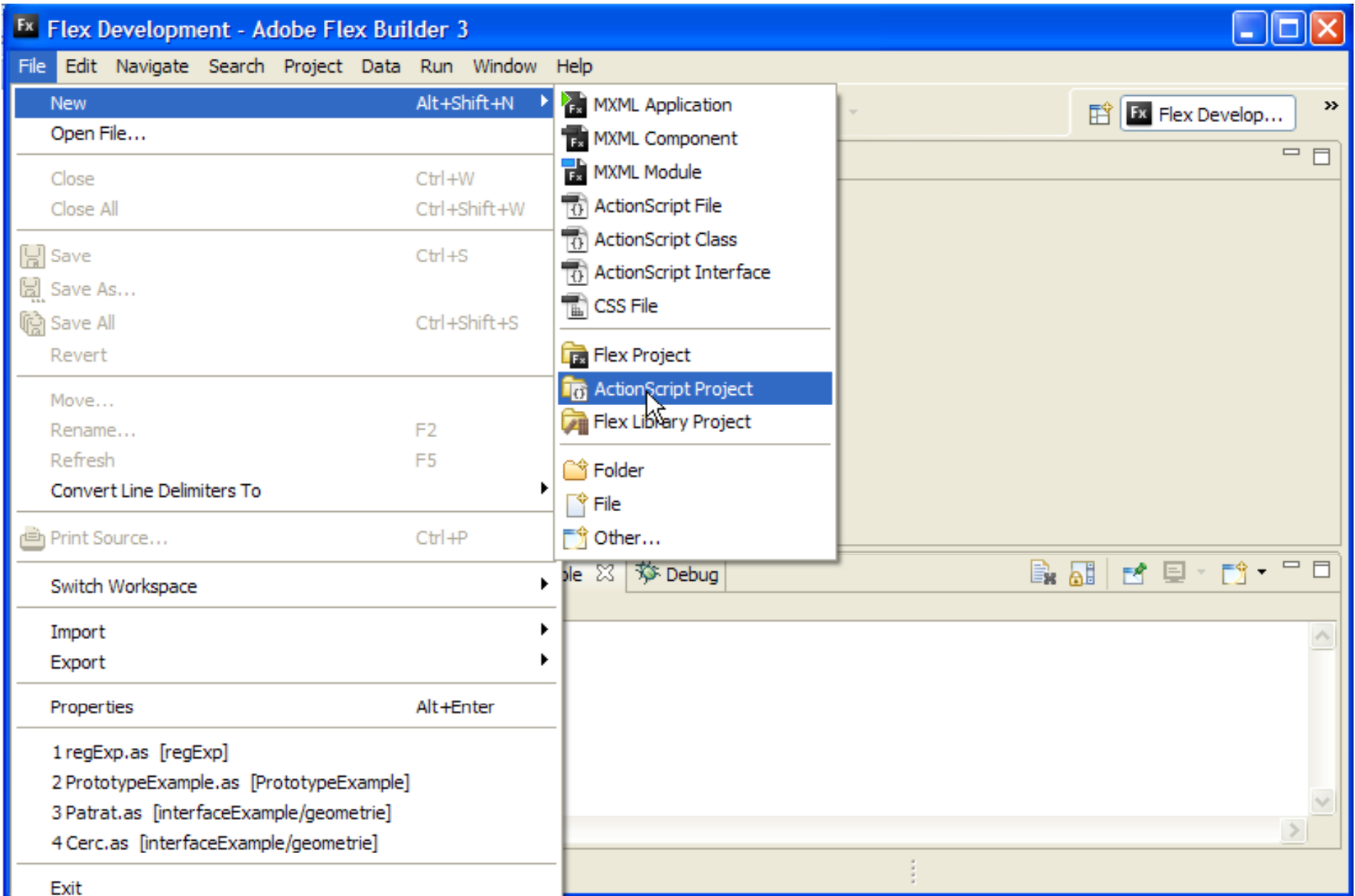
Adauga o serie de biblioteci suplimentare flexului pentru a permite crearea de aplicatii desktop utilizand AS3.

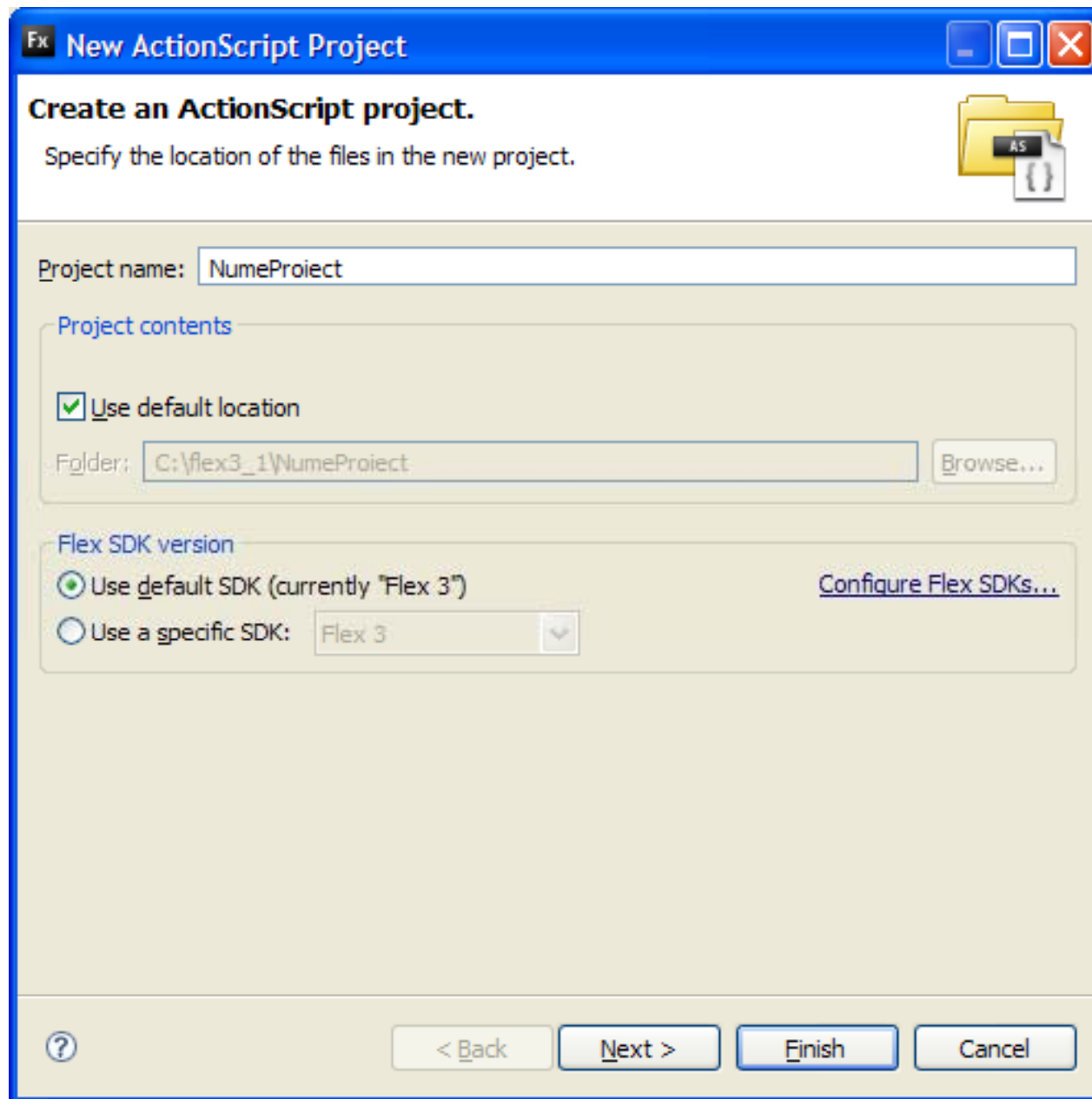
Ce vom utiliza ?

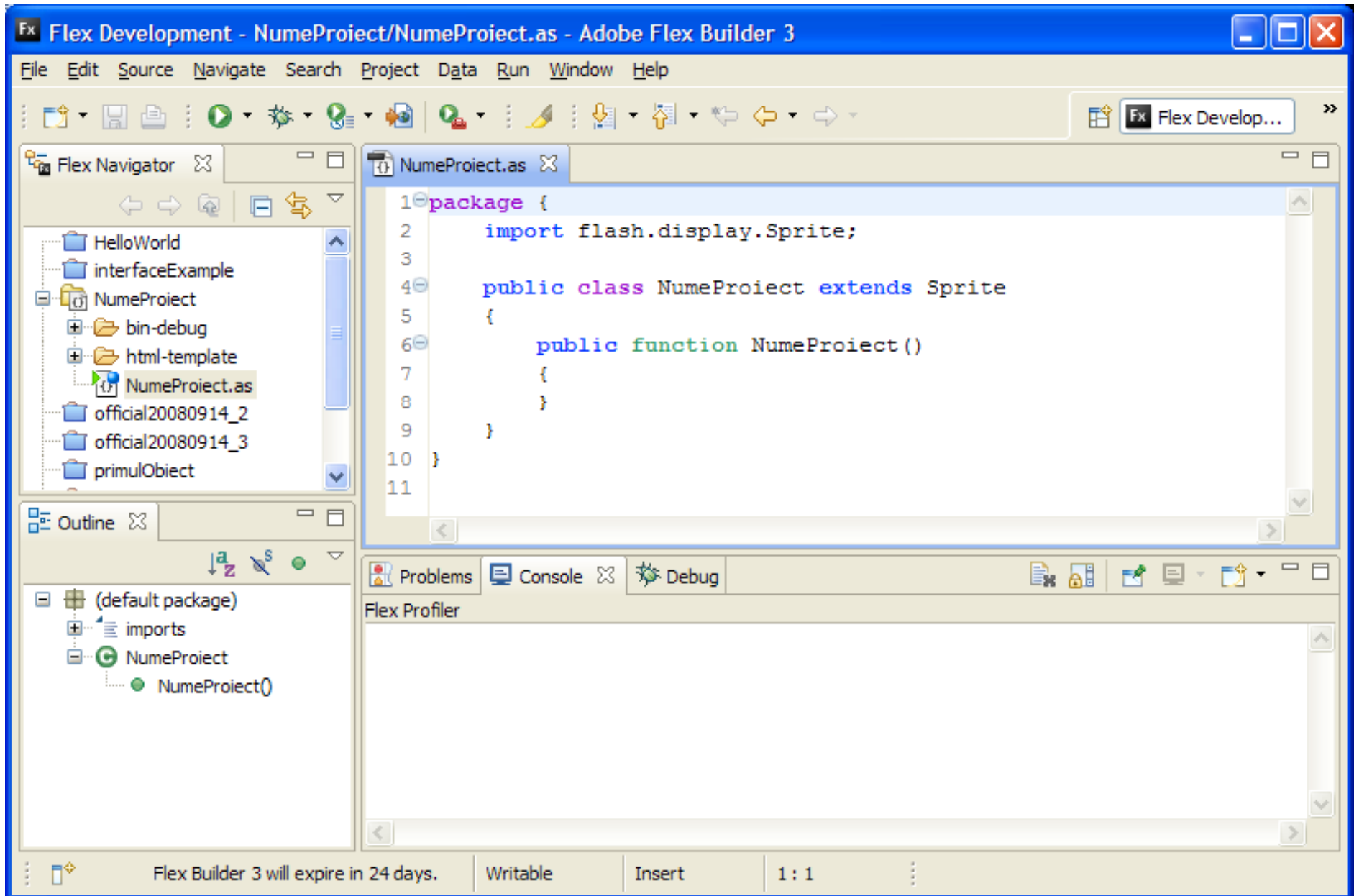
Ce vom utiliza ?











```
package {  
    import flash.display.Sprite;  
  
    public class NumeProiect extends Sprite  
    {  
        public function NumeProiect()  
        {  
        }  
    }  
}
```

Eu sunt o clasa

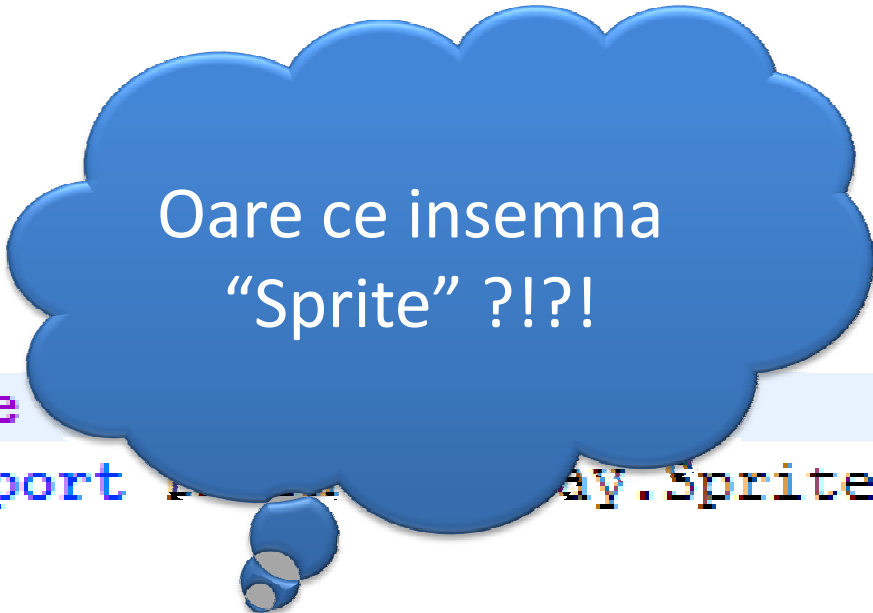
```
package  
import  
public class NumeProiect extends Sprite  
{  
    public function NumeProiect()  
    {  
    }  
}
```

Am numele
proiectului si sunt
salvata in fisierul
"NumeProiect.as"

```
package {  
    import flash.  
  
    public class NumeProiect extends Sprite  
    {  
        public function NumeProiect()  
        {  
        }  
    }  
}
```

Si aproape mereu
voi extinde clasa
"Sprite"

```
package {  
    import flash.display.*  
  
    public class NumeProiect extends Sprite  
    {  
        public function NumeProiect()  
        {  
        }  
    }  
}
```



```
package  
import com.badlogic.gdx.graphics.Sprite;  
  
public class NumeProiect extends Sprite  
{  
    public function NumeProiect()  
    {  
    }  
}
```

Imi zice aceasta comanda (care e la fel ca include din C, C++)

```
package {  
    import flash.display.Sprite;  
  
    public class NumeProiect extends Sprite  
    {  
        public function NumeProiect()  
        {  
        }  
    }  
}
```

Impreuna cu mai multe clase ce-mi seamana, pot forma un pachet

```
package {  
    import flash.  
  
    public class NumeProiect extends Sprite  
    {  
        public function NumeProiect()  
        {  
        }  
    }  
}
```

Daca as face parte dintr-un pachet, as fi trecut si numele acestuia...

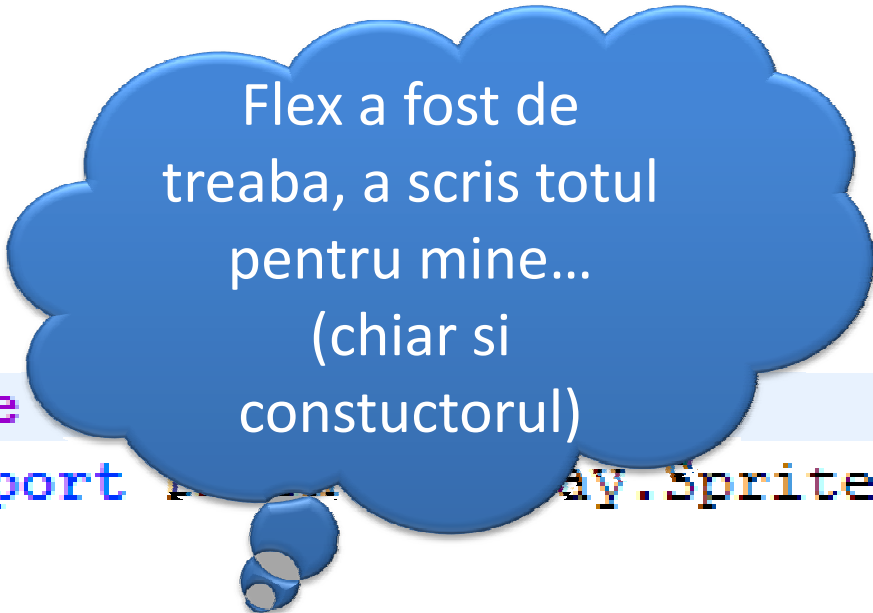
```
package {  
    import flash.display.*
```

```
public class NumeProiect extends Sprite  
{  
    public function NumeProiect()  
    {  
    }  
}
```

```
}
```

La fel ca orice clasa
(Din C++, Java, etc)
am un constructor

```
package {
import
public ends Sprite
{
    public function NumeProiect ()
    {
    }
}
}
```



```
package  
import flash.display.Sprite;  
  
public class NumeProiect extends Sprite  
{  
    public function NumeProiect()  
    {  
    }  
}
```

Constructorul este
functia pe care o
execut de fiecare
data cand sunt
instantiata

```
package {  
    import  
  
    public  
    {  
        public function NumeProiect()  
        {  
        }  
    }  
}
```

```
package {  
    import flash.display.  
  
    public class NumeP  
    {  
        public function NumeProiect()  
        {  
        }  
    }  
}
```

Constructorul are mereu acelasi nume ca si clasa din care face parte...

Prima noastra variabilaaaaaaa... 😊

```
package {  
    import flash.display.Sprite;  
  
    public class NumeProiect extends Sprite  
    {  
        public function NumeProiect()  
        {  
            var mesaj:String;  
        }  
    }  
}
```

Salut, eu sunt o variabila !
(de tip String)

Prima noastra variabilaaaaaaa... 😊

```
package {  
    import flash.display.Sprite;  
  
    public class NumeProiect extends Sprite  
    {  
        public function NumeProiect()  
        {  
            var mesaj:String;  
        }  
    }  
}
```

Din cauza ca sunt de tip String, pot tine minte "texte"

Prima noastra variabilaaaaaaa... 😊

```
package {  
    import flash.display.Sprite;  
  
    public class NumeProiect extends Sprite  
    {  
        public function NumeProiect()  
        {  
            var mesaj:String;  
        }  
    }  
}
```

Mi-ar place sa am
valoarea:
"Hello World"

Prima noastra variabilaaaaaaa... 😊

```
package {  
    import flash.display.Sprite;  
  
    public class NumeProiect extends Sprite  
    {  
        public function NumeProiect()  
        {  
            var mesaj:String;  
            mesaj = "Hello World";  
        }  
    }  
}
```

done...

Primul textfield



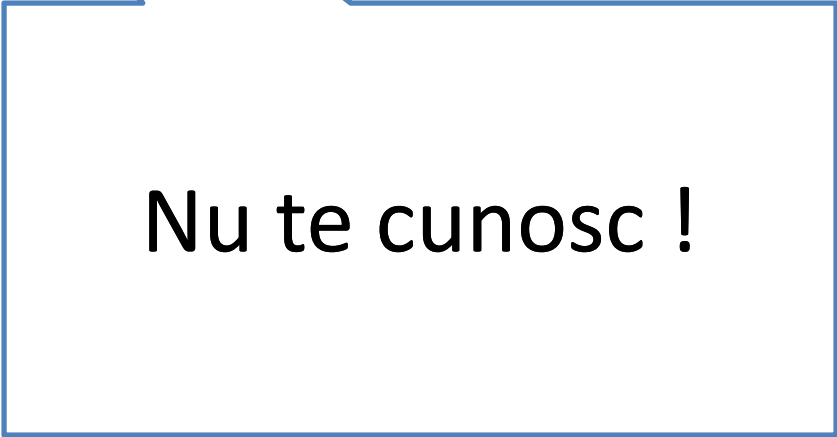
```
public class NumeProiect extends Sprite
{
    public function NumeProiect()
    {
        var mesaj:String;
        mesaj = "Hello World";
        var textField:TextField;
    }
}
```

Eu sunt un obiect de tip TextField si pot afisa ceva intr-o zona ecran

Primul textfield



```
public class NumeProiect extends Sprite
{
    public function NumeProiect()
    {
        var mesaj:String;
        mesaj = "Hello World";
        var textField:Textfield;
    }
}
```



Nu te cunosc !

Primul textfield

O sa marchez linia pe care esti cu rosu (sa te vada programatoru')

```
1 package {
2     import flash.displ
3
4     public class NumeProiect extends Sprite
5     {
6         public function NumeProiect()
7         {
8             var mesaj:String;
9             mesaj = "Hello World";
10            var textField:TextField;
11        }
12    }
13 }
14
```

Primul textfield

```
package {
    import flash.display.Sprite;
    import flash.text.TextField;

    public class NumeProiect extends Sprite
    {
        public function NumeProiect()
        {
            var mesaj:String;
            mesaj = "Hello World";
            var textField:TextField;
        }
    }
}
```

Paiii.. Sa-ti dau cartea mea de vizita:
import flash.text.TextField;

Primul textfield

```
package {  
    import flash.display.Sprite;  
    import flash.text.TextField;  
  
    public class NumeProiect extends Sprite  
    {  
        public function NumeProiect()  
        {  
            var mesaj : String;  
            mesaj = "Hello World";  
            var textField : TextField;  
        }  
    }  
}
```

Toate importurile se fac inainte de declararea clasei

Primul textfield

```
package {
    import flash.display.Sprite;
    import flash.text.TextField;

    public class NumeProiect extends Sprite
    {
        public function NumeProiect()
        {
            var mesaj:String;
            mesaj = "Hello World";
            var textField:TextField;
        }
    }
}
```

Perfect, acum stiu totul despre tine... [ce poti sa faci, ce proprietati ai ... TOT]

Primul textfield

```
package {  
    import flash.display.Sprite;  
    import flash.text.TextField;  
  
    public class NumeProiect extends Sprite  
    {  
        public function NumeProiect()  
        {  
            var mesaj:String;  
            mesaj = "Hello World";  
            var textField:TextField;  
        }  
    }  
}
```

Pentru a putea exista pe ecran, am nevoie de fiu instantiat apoi adaugat in lista obiectelor afisate...

Primul textfield

```
package {
    import flash.display.Sprite;
    import flash.text.TextField;

    public class NumeProiect extends Sprite
    {
        public function NumeProiect()
        {
            var mesaj:String;
            mesaj = "Hello World";
            var textField:TextField;
        }
    }
}
```

De fapt eu eXist ca variabila dar nu am o zona de memorie in care sa imi tin "catrafusele"

Primul textfield

```
package {
    import flash.display.Sprite;
    import flash.text.TextField;

    public class NumeProiect extends Sprite
    {
        public function NumeProiect()
        {
            var mesaj:String;
            mesaj = "Hello World";
            var textField:TextField;
        }
    }
}
```

Operatorul “new” poate crea un nou obiect in memorie si daca voi fi atribuit acestui obiect.. ar fi PERFECT

Primul textfield

```
package {
    import flash.display.Sprite;
    import flash.text.TextField;

    public class NumeProiect extends Sprite
    {
        public function NumeProiect()
        {
            var mesaj:String;
            mesaj = "Hello World";
            var textField:TextField = new TextField();
        }
    }
}
```

Pe langa a crea un obiect, eu apelez si constructorul acestuia... adica TextField()

Primul textfield

```
package {  
    import flash.display.Sprite;  
    import flash.text.TextField;  
  
    public class NumeProiect extends Sprite  
    {  
        public function NumeProiect()  
        {  
            var mesaj:String;  
            mesaj = "Hello World";  
            var textField:TextField = new TextField();  
        }  
    }  
}
```

Cum ramane cu
adaugarea pe ecran ?!?!

```
package {
    import flash.display.Sprite;
    import flash.text.TextField;

    public class NumeProiect extends Sprite
    {
        public function NumeProiect()
        {
            var mesaj : String;
            mesaj = "Hello World";
            var textField : TextField = new TextField();
            addChild( textField );
        }
    }
}
```

Rezolv eu astaaaaa...

```
package {
    import flash.display.Sprite;
    import flash.text.TextField;

    public class NumeProiect extends Sprite
    {
        public function NumeProiect()
        {
            var mesaj:String;
            mesaj = "Hello World";
            var textField:TextField;
            addChild( textField );
        }
    }
}
```

Da tu cine esti ?!

```
package {  
    import flash.display.Sprite;  
    import flash.text.TextField;  
  
    public class NumeProiect extends Sprite  
    {  
        public function NumeProiect ()  
        {  
            var mesaj:String;  
            mesaj = "Hello World";  
            var textField:TextField;  
            addChild( textField );  
        }  
    }  
}
```

O metoda a clasei Sprite pe care tu o extinzi...
daca extinzi o clasa, mostenesti toate proprietatile
si metodele acelei clase... ai uitat deja ?!

Scuze, ai dreptate... de fapt eu insumi sunt un Sprite si e firesc sa cunosc metoda addChild (pentru ca este a mea)

```
package {
    import flash.display.Sprite;
    import flash.text.TextField;

    public class NumeProiect extends Sprite
    {
        public function NumeProiect()
        {
            var mesaj:String;
            mesaj = "Hello World";
            var textField:TextField;
            addChild( textField );
        }
    }
}
```

```
package {  
    import flash.display.Sprite;  
    import flash.text.TextField;  
  
    public class NumeProiect extends Sprite  
    {  
        public function NumeProiect()  
        {  
            var mesaj:String;  
            mesaj = "Hello World";  
            var textField:TextField;  
            addChild( textField );  
        }  
    }  
}
```

Mulumesc addChild, acum sunt si eu pe ecran. Imi puteti completa proprietatea "text" pentru a afisa ceva... altfel sunt inutil

```
package {  
    import flash.display.Sprite;  
    import flash.text.TextField;  
  
    public class NumeProiect extends Sprite  
    {  
        public function NumeProiect()  
        {  
            var mesaj:String;  
            mesaj = "Hello World";  
            var textField:TextField;  
            addChild( textField );  
        }  
    }  
}
```

Proprietatea “text” este de tip String, eu voi afisa imediat textul dat ca valoare... Ati putea face ceva de genul: textField.text = mesaj;

```
package {
    import flash.display.Sprite;
    import flash.text.TextField;

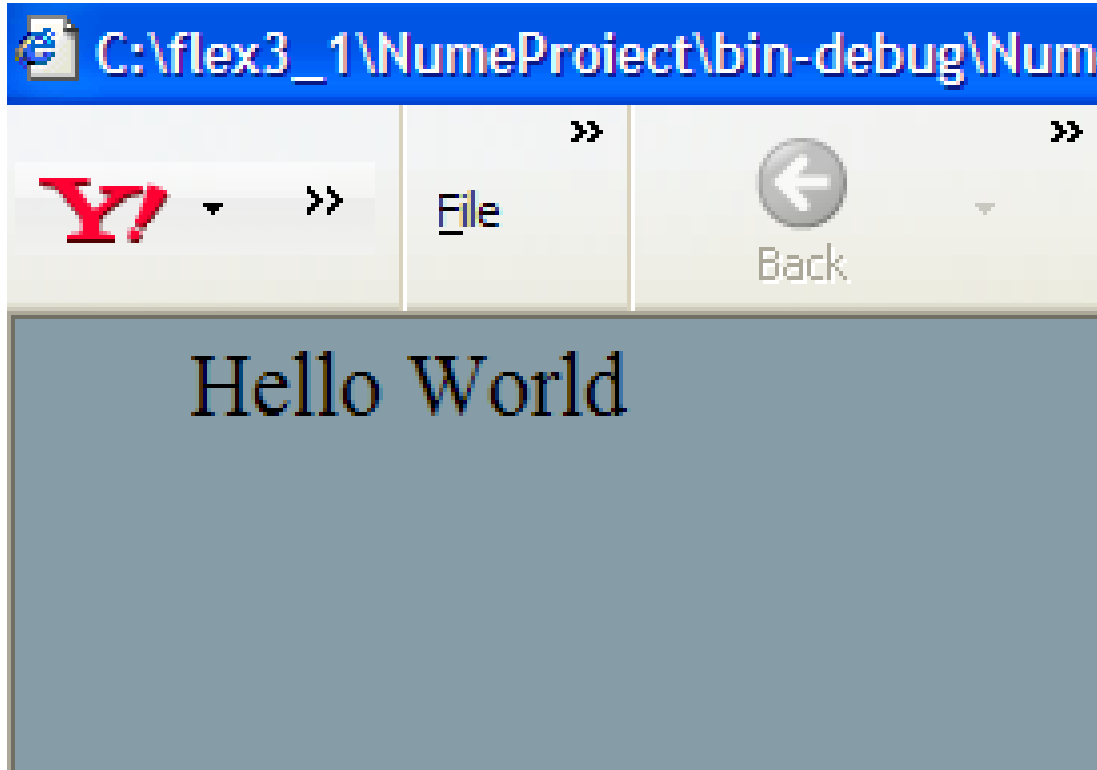
    public class NumeProiect extends Sprite
    {
        public function NumeProiect()
        {
            var mesaj:String;
            mesaj = "Hello World";
            var textField:TextField;
            addChild( textField );
        }
    }
}
```

```
package {  
    import flash.display.Sprite;  
    import flash.text.TextField;  
  
    public class NumeProiect extends Sprite  
    {  
        public function NumeProiect()  
        {  
            var mesaj : String;  
            mesaj = "Hello World";  
            var textField : TextField = new TextField();  
            addChild( textField );  
            textField.text = mesaj;  
        }  
    }  
}
```

Asa mai merge...

```
package {  
    import flash.display.Sprite;  
    import flash.text.TextField;  
  
    public class NumeProiect extends Sprite  
    {  
        public function NumeProiect()  
        {  
            var mesaj : String;  
            mesaj = "Hello World";  
            var textField : TextField = new TextField();  
            addChild( textField );  
            textField.text = mesaj;  
        }  
    }  
}
```

OK, ar fi timpul sa
facem o compilare: F11



```
package {
    import flash.display.Sprite;
    import flash.text.TextField;

    public class NumeProiect extends Sprite
    {
        public function NumeProiect()
        {
            var mesaj : String;
            mesaj = "Hello World";
            var textField : TextField = new TextField();
            addChild( textField );
            textField.text = mesaj;
        }
    }
}
```

O singura chestie cred ca am uitat sa zic.. Ce inseamna “public” care apare ici si colo...

```
package {
    import flash.display.Sprite;
    import flash.text.TextField;

    public class NumeProiect extends Sprite
    {
        public function NumeProiect ()
        {
            var mesaj : String;
            mesaj = "Hello World";
            var textField : TextField = new TextField();
            addChild( textField );
            textField.text = mesaj;
        }
    }
}
```

Cand pun "public" in fata unei metode sau proprietati, voi face acea met/prop sa poata fi accesata si de alte obiecte nu numai de mine

```
package {  
    import flash.display.Sprite;  
    import flash.text.TextField;  
  
    public class NumeProiect extends Sprite  
    {  
        public function NumeProiect()  
        {  
            var mesaj : String;  
            mesaj = "Hello World";  
            var textField : TextField = new TextField();  
            addChild( textField );  
            textField.text = mesaj;  
        }  
    }  
}
```

Eu nu am “public”... de ce ?!?!

```
package {
    import flash.display.Sprite;
    import flash.text.TextField;

    public class NumeProiect extends Sprite
    {
        public function NumeProiect ()
        {
            var mesaj : String;
            mesaj = "Hello World";
            var textField : TextField = new TextField();
            addChild(textField);
            textField.text = mesaj;
        }
    }
}
```

Variabilele sau functiile din cadrul metodelor nu pot avea "public" sau nimic altceva. Ele vor fi accesibile numai in cadrul metodei in care au definite (adica scopul lor este numai acea metoda)

```
package {  
    import flash.display.Sprite;  
    import flash.text.TextField;  
  
    public class NumeProiect extends Sprite  
    {  
        public function NumeProiect()  
        {  
            var mesaj : String;  
            mesaj = "Hello World";  
            var textField : TextField = new TextField();  
            addChild( textField );  
            textField.text = mesaj;  
        }  
    }  
}
```

Mai exista si altceva inafara de "public" ?!

```
package {  
    import flash.display.Sprite;  
    import flash.text.TextField;  
  
    public class NumeProiect extends Sprite  
    {  
        public function NumeProiect()  
        {  
            var mesaj:String;  
            mesaj = "Hello World";  
            var textField:TextField = new TextField();  
            addChild(textField);  
            textField.text = mesaj;  
        }  
    }  
}
```

Da: public, private, protected, internal... Aceste cuvinte cheie se numesc **“Modificatori de acces”**

```
package {
    import flash.display.Sprite;
    import flash.text.TextField;

    public class NumeProiect extends Sprite
    {
        public function NumeProiect()
        {
            var mesaj : String;
            mesaj = "Hello World";
            var textField : TextField = new TextField();
            addChild( textField );
            textField.text = mesaj;
        }
    }
}
```

Niceee... ce inseamna fiecare ?:D

```
package {  
    import flash.display.Sprite;  
    import flash.text.TextField;  
  
    public class NumeProiect extends Sprite  
    {  
        public function NumeProiect()  
        {  
            var mesaj:String;  
            mesaj = "Hello World";  
            var textField:TextField = new TextField();  
            addChild(textField);  
            textField.text = mesaj;  
        }  
    }  
}
```

public: accesibil de oriunde;
private: accesibil numai in clasa curenta;
protected: accesibil in clasa curenta si in cele ce o extind;
internal: accesibil numai in cadrul pachetului

```
package {  
    import flash.display.Sprite;  
    import flash.text.TextField;  
  
    public class NumeProiect extends Sprite  
    {  
        public function NumeProiect()  
        {  
            var mesaj : String;  
            mesaj = "Hello World";  
            var textField : TextField = new TextField();  
            addChild( textField );  
            textField.text = mesaj;  
        }  
    }  
}
```

Vreau si eu un modificador de acces, ce-mi recomanzi ?!

```
package {
    import flash.display.Sprite;
    import flash.text.TextField;

    public class NumeProiect extends Sprite
    {
        public function NumeProiect()
        {
            var mesaj:String;
            mesaj = "Hello World";
            var textField:TextField = new TextField();
            addChild( textField );
            textField.text = mesaj;
        }
    }
}
```

- Deoarece alte clase nu cred ca le intereseaza ce reprezinti, recomand **private**...
Numai ca trebuie sa te scot inafara constructorului , esti de acord?

```
package {
    import flash.display.Sprite;
    import flash.text.TextField;

    public class NumeProiect extends Sprite
    {
        public function NumeProiect()
        {
            var mesaj : String;
            mesaj = "Hello World";
            var textField : TextField = new TextField();
            addChild( textField );
            textField.text = mesaj;
        }
    }
}
```

DA !

```
package {
    import flash.display.Sprite;
    import flash.text.TextField;

    public class NumeProiect extends Sprite
    {
        private var mesaj : String;
        public function NumeProiect ()
        {
            mesaj = "Hello World";
            var textField : TextField = new TextField();
            addChild( textField );
            textField.text = mesaj;
        }
    }
}
```



```
package {
    import flash.display.Sprite;
    import flash.text.TextField;

    public class NumeProiect extends Sprite
    {
        private var mesaj : String;
        public function NumeProiect ()
        {
            mesaj = "Hello World";
            var textField : TextField = new TextField();
            addChild( textField );
            textField.text = mesaj;
        }
    }
}
```

Acum pot fi accesat si din alte metode din aceasta clasa, nu numai din constructor, nu ?

```
package {  
    import flash.display.Sprite;  
    import flash.text.TextField;  
  
    public class NumeProiect extends Sprite  
    {  
        private var mesaj : String;  
        public function NumeProiect ()  
        {  
            mesaj = "Hello World";  
            var textField : TextField = new TextField();  
            addChild( textField );  
            textField.text = mesaj;  
        }  
    }  
}
```

Da 😊

```
package {
    import flash.display.Sprite;
    import flash.text.TextField;

    public class NumeProiect extends Sprite
    {
        private var mesaj : String;
        public function NumeProiect ()
        {
            mesaj = "Hello World";
            var textField : TextField = new TextField();
            addChild(textField);
            textField.text = mesaj;
        }
    }
}
```

In cazul in care modificatorul de acces lipseste pentru o metoda sau proprietate, este considerata public

Recapitulare... & Coddling...

```
package {
    import flash.display.Sprite;
    import flash.text.TextField;

    public class NumeProiect extends Sprite
    {
        private var mesaj : String;
        public function NumeProiect()
        {
            mesaj = "Hello World";
            var textField : TextField = new TextField();
            addChild( textField );
            textField.text = mesaj;
        }
    }
}
```

Creati un nou proiect: Interactiune

- Dati click dreapta pe numele proiectului, selectati optiunea **New** apoi **Folder**
- Noul director se va numi “classes”
- Dati click dreapta pe directorul classes si selectati optiunea **New** apoi **ActionScript Class**
- Denumiti clasa “Minge” dupa care apasati Finish.
- Proiectul va contine doua clase in fisierele: **Interactiune.as** si **Minge.as** (ultimul aflat in directorul “classes”)

Interactiune.as – pasul 1

```
package {  
    import classes.Minge;  
    import flash.display.Sprite;  
  
    public class Interactiune extends Sprite  
    {  
        private var minge:Minge = new Minge();  
        public function Interactiune()  
        {  
            addChild( minge );  
            minge.x = 0;  
            minge.y = 200;  
        }  
    }  
}
```

Interactiune.as – pasul 1

```
package {  
    import classes.Minge;  
    import flash.display.Sprite;  
  
    public class Interactiune extends Sprite  
    {  
        private var minge:Minge = new Minge();  
        public function Interactiune()  
        {  
            addChild(minge);  
            minge.x = 100;  
            minge.y = 100;  
        }  
    }  
}
```

Asa se importa o clasa din pachetul classes...

Interactiune.as – pasul 1

```
package {  
    import classes.Minge;  
    import flash.display.Sprite;  
  
    public class Interactiune extends Sprite  
    {  
        private var minge:Minge = new Minge();  
        public function Interactiune()  
        {  
            addChild( minge );  
            minge.x = 0;  
            minge.y = 20;  
        }  
    }  
}
```

Facem un obiect de tip Minge

Interactiune.as – pasul 1

```
package {  
    import classes.Minge;  
    import flash.  
  
    public class  
    {  
        private var minge : Minge = new Minge();  
        public function interactiune()  
        {  
            addChild( minge );  
            minge.x = 0;  
            minge.y = 200;  
        }  
    }  
}
```

Il adaugam in lista obiectelor ce vor fi afisate

Interactiune.as – pasul 1

```
package {  
    import classes  
    import flash.d  
  
    public class I  
    {  
        private var minge = new Minge();  
        public functia Interactiune()  
        {  
            addChild( minge );  
            minge.x = 0;  
            minge.y = 200;  
        }  
    }  
}
```

Minge va extinde sprite si din cauza asta are proprietatile x, y = pozitia pe ecran...

Interactiune.as – pasul 1

```
package {  
    import classes.Minge;  
    import flash.display.Sprite;  
  
    public class Interactiune extends Sprite  
    {  
        private var minge:Minge = new Minge();  
        public function Interactiune()  
        {  
            addChild( minge );  
            minge.x = 0;  
            minge.y = 200;  
        }  
    }  
}
```

Minge.as – pas1

```
package classes
{
    import flash.display.Sprite;

    public class Minge extends Sprite
    {
        public function Minge()
        {
            graphics.beginFill( 0xff0000, 1);
            graphics.drawCircle( -20, -20, 40 );
        }
    }
}
```

Minge.as – pas1

```
package classes
{
    import flash.display.Sprite;

    public class Minge extends Sprite
    {
        public function Minge()
        {
            graphics.beginFill(0xff0000, 1);
            graphics.drawCircle(0, 0, 20);
        }
    }
}
```

Aceasta clasa nu mai extinde automat Sprite dar putem noi face acest lucru:

Minge.as – pas1

```
package classes
{
    import flash.display.Sprite;

    public class Minge extends Sprite
    {
        public function Minge()
        {
            graphics.beginFill( 0xff0000, 1);
            graphics.drawCircle( -20, -20, 40 );
        }
    }
}
```

Daca am scris ca extinde Sprite,
Flex-ul ne ajuta si scrie el singur
importul

Minge.as – pas1

```
package classes
{
    import flash.display.Sprite;

    public class Minge extends Sprite
    {
        public function Minge()
        {
            graphics.beginFill( 0xff0000, 1);
            graphics.drawCircle( -20, -20, 40 );
        }
    }
}
```

Vom face o figura ce va fi colorata cu rosu (0xff0000) si va fi opaca (opacitatea = 1 inseamna complet opaca)

Minge.as – pas1

```
package classes
{
    import flash.display.Sprite;

    public class Minge extends Sprite
    {
        public function Minge()
        {
            graphics.beginFill( 0xff0000, 1);
            graphics.drawCircle( -20, -20, 40 );
        }
    }
}
```

Figura va fi un cerc.
Deoarece acestea sunt scrise in constructor, vor fi apelate de new Minge() – din Interactiune.as

Minge.as – pas1

```
package classes
{
    import flash.display.Sprite;

    public class Minge extends Sprite
    {
        public function Minge()
        {
            graphics.beginFill( 0xff0000, 1);
            graphics.drawCircle( -20, -20, 40 );
        }
    }
}
```

F11

Minge.as – pas2 (adaugarea miscarii)

```
package classes
{
    import flash.display.Sprite;
    import flash.utils.setInterval;

    public class Minge extends Sprite
    {
        public function Minge()
        {
            graphics.beginFill( 0xff0000, 1);
            graphics.drawCircle( -20, -20, 40 );
            setInterval( misca, 10 );
        }
        private function misca():void
        {
            this.x += 1;
        }
    }
}
```

Minge.as – pas2 (adaugarea miscarii)

```
package classes
{
    import flash.display.Sprite;
    import flash.utils.setInterval;

    public
    {
        public
        {
            graphics.beginFill( 0xff0000, 1);
            graphics.drawCircle( -20, -20, 40 );
            setInterval( misca, 10 );
        }
        private function misca():void
        {
            this.x += 1;
        }
    }
}
```

La fiecare 10ms va fi apelata metoda misca

Minge.as – pas2 (adaugarea miscarii)

```
package classes
{
    import flash.display.Sprite;
    import flash.utils.setInterval;

    public class Minge extends Sprite
    {
        public function Minge()
        {
            graphics.drawCircle( -20, -20, 40 );
            setInterval( misca, 10 );
        }
        private function misca():void
        {
            this.x += 1;
        }
    }
}
```

Importul pentru setInterval este facut automat de Flex

Minge.as – pas2 (adaugarea miscarii)

```
package classes
{
    import flash.display.Sprite;
    import flash.utils.setInterval;

    public class Minge
    {
        public function Minge()
        {
            graphics.drawCircle( -20, -20, 40 );
            setInterval( misca, 10 );
        }
        private function misca():void
        {
            this.x += 1;
        }
    }
}
```

Functia "misca" va deplasa putin mai in dreapta obiectul curent

Minge.as – pas2 (adaugarea miscarii)

```
package classes
{
    import flash.display.Sprite;
    import flash.utils.setInterval;

    public class Minge
    {
        public function Minge()
        {
            graphics.drawCircle( -20, -20, 40 );
            setInterval( misca, 10 );
        }
        private function misca():void
        {
            this.x += 1;
        }
    }
}
```

Va fi accesibila numai din aceasta clasa...

Minge.as – pas2 (adaugarea miscarii)

```
package classes  
{  
    import flash.display.Sprite;  
    import flash.utils.setInterval;
```

```
public class Minge  
{  
    public Minge()  
    {  
        graphic = new Sprite( -20, -20, 40 );  
        setInterval( misca, 10 );  
    }  
    private function misca():void  
    {  
        this.x += 1;  
    }  
}
```

Si va modifica pozitia pe X a mingei

Minge.as – pas2 (adaugarea miscarii)

```
package classes  
{
```

Tot aici am putea face ceva de genu:
if (this.x > 400) this.x = 0;
(in acest fel mingea nu ar mai parasi ecranul)

```
    g = new Circle( -20, -20, 40 );  
    setMovement( misca, 10 );  
}  
private function misca():void  
{  
    this.x += 1;  
}  
}
```

Minge.as – pas2 (adaugarea miscarii)

```
package classes
{
    import flash.display.Sprite;
    import flash.utils.setInterval;

    public class Minge extends Sprite
    {
        public function Minge()
        {
            graphics.beginFill( 0xff0000, 1);
            graphics.drawCircle( -20, -20, 40 );
            setInterval( misca, 10 );
        }
        private function misca():void
        {
            this.x += 1;
        }
    }
}
```

Interactiune.as – pasul 2

```
package {  
    import classes.Minge;  
    import flash.display.Sprite;  
    import flash.events.MouseEvent;  
  
    public class Interactiune extends Sprite  
    {  
        private var minge:Minge = new Minge();  
        public function Interactiune()  
        {  
            addChild( minge );  
            minge.x = 0;  
            minge.y = 200;  
            minge.addEventListener(MouseEvent.CLICK, resetPos);  
        }  
  
        private function resetPos(e:MouseEvent):void  
        {  
            e.target.x = 0;  
        }  
    }  
}
```

Interactiune.as – pasul 2

```
package {
    import classes.Minge;
    import flash.display.Sprite;
    import flash.events.MouseEvent;

    public class Interactiune extends Sprite
    {
        private var minge : Minge;

        public function Interactiune()
        {
            addChild(minge);
            minge.x = 0;
            minge.y = 200;
            minge.addEventListener(MouseEvent.CLICK, resetPos);
        }

        private function resetPos(e:MouseEvent):void
        {
            e.target.x = 0;
        }
    }
}
```

Urmărim cand se
apasa pe minge

Interactiune.as – pasul 2

```
package {  
    import classes.Minge;  
    import flash.display.Sprite;  
    import flash.events.MouseEvent;  
  
    public class Interactiune extends Sprite  
    {  
        private var minge:Minge = new Minge();  
        public function Interactiune() {  
            addChild( minge );  
            minge.x = 0;  
            minge.y = 200;  
            minge.addEventListener(MouseEvent.CLICK, resetPos);  
        }  
  
        private function resetPos(e:MouseEvent):void  
        {  
            e.target.x = 0;  
        }  
    }  
}
```

Si apelam metoda
resetPos cand se
intampla acest lucru

Interactiune.as – pasul 2

```
package {  
    import classes.Minge;  
    import flash.display.Sprite;  
    import flash.events.MouseEvent;  
  
    public class Interactiune extends Sprite  
    {  
        private var minge:Minge = new Minge();  
        public function Interactiune() {  
            addChild( minge );  
            minge.x = 0;  
            minge.y = 200;  
            minge.addEventListener(MouseEvent.CLICK, resetPos);  
        }  
  
        private function resetPos(e:MouseEvent):void  
        {  
            e.target.x = 0;  
        }  
    }  
}
```

Care duce mingea
apelanta in pozitia
initiala

Interactiune.as – pasul 2

```
package {
    import classes.Minge;
    import flash.display.Sprite;
    import flash.events.MouseEvent;

    public class Interactiune extends Sprite
    {
        private var minge:Minge = minge;

        public function Interactiune()
        {
            addChild( minge );
            minge.x = 0;
            minge.y = 200;
            minge.addEventListener(MouseEvent.CLICK, click);
        }

        private function resetPos(e:MouseEvent):void
        {
            e.target.x = 0;
        }
    }
}
```

Importul pentru eveniment este facut iarasi automat (de Flex)

Interactiune.as – pasul 2

```
package {
    import classes.Minge;
    import flash.display.Sprite;
    import flash.events.MouseEvent;

    public class Interactiune extends Sprite
    {
        private var minge:Minge = new Minge();
        public function Interactiune()
        {
            addChild( minge );
            minge.x = 0;
            minge.y = 200;
            minge.addEventListener(MouseEvent.CLICK, resetPos);
        }

        private function resetPos(e:MouseEvent):void
        {
            e.target.x = 0;
        }
    }
}
```

Interactiune.as – pasul 3

```
package {
    import classes.Minge;
    import flash.display.Sprite;
    import flash.events.MouseEvent;

    public class Interactiune extends Sprite
    {
        private var sirMingi:Array = new Array();
        public function Interactiune()
        {
            for(var i:uint = 0; i<20; i++)
            {
                var minge:Minge = new Minge();
                sirMingi.push( minge );
                minge.x = Math.random() * 500;
                minge.y = Math.random() * 400;
                minge.addEventListener(MouseEvent.CLICK, resetPos);
                addChild( minge );
            }
        }
        private function resetPos(e:MouseEvent):void
        {
            e.target.x = 0;
        }
    }
}
```

Tema...

- Modificati ultima aplicatie astfel:
 - Constuiti in **minge.as** o variabila privata de tip Number cu numele "speed" a carei valoare va fi setata aleator (asa cum am setat in Interactiune.as pozitia bilelor)
 - In **Interactiune.as** introduceti un textField ce va afisa un scor ce va fi incrementat de fiecare data cand o minge este apasata
 - Nu lasati bilele sa iasa din ecran (este dat un pont in acest sens pe unul din slidurile anterioare)

Intrebari ?!?!