

## Tutorial 11 - Utilització de components

### Pas 1 de 14

Un component és un objecte reutilitzable que pot incloure gràfics i codi preconfigurats, i els paràmetres del qual els podem modificar fàcilment.

Utilitzarem components per construir una minigaleria d'imatges i un camp de text.

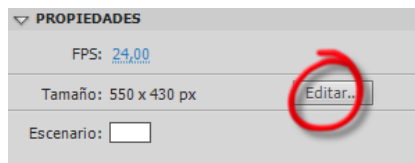
Les imatges i el text que utilitzarem en aquest tutorial es troben en l'arxiu [recursosTutorial11.zip](#).

Aquest arxiu conté una carpeta anomenada *images* que haurem de posar en la mateixa carpeta en la qual tinguem el nostre arxiu Flash.



### Pas 2 de 14

Canviem la mida de l'escenari a 550 x 430 píxels.



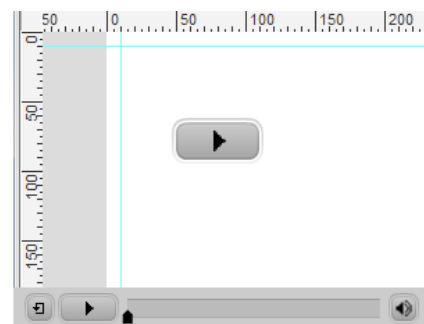
Entre els elements que posem en l'escenari i els límits del document deixarem un marge de 10 píxels.

Ens podem ajudar de guies per facilitar el posicionament d'objectes en punts específics de l'escenari.

Seleccionem **Ver > Reglas** i ens assegurem que la casella **Ver > Guías > Mostrar guías** estigui seleccionada.

Fem clic sobre les regles i arrosseguem per posicionar quatre guies en els llocs que necessitem, en aquest cas a 10 i 420 píxels en les guies horitzontals, i a 10 i 540 en les verticals.

Per col·locar les guies en aquestes posicions podem arrossegar-les fins a una posició aproximada, i després podem introduir el valor exacte fent doble clic sobre elles.



### Pas 3 de 14

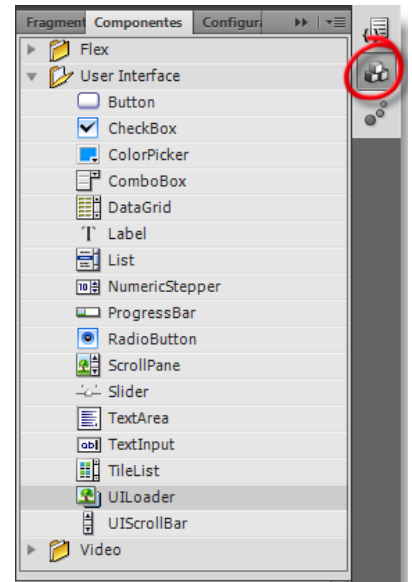
El primer component que utilitzarem és **UILoader**, un contenidor que permet carregar amb facilitat imatges o arxius swf externs a l'aplicació.

Obrim el panell **Componentes**, i en la carpeta **User Interface** fem doble clic sobre el component **UILoader**. En el centre de l'escenari apareixerà una instància d'aquest component.

Utilitzarem aquest component per carregar l'imatge principal de la nostra galeria. En l'inspector de *Propiedades* hi assignem una mida de **AN:400** i **AL:300**, que és la mida de les nostres imatges.



Posicionem el contenidor en X:10 i Y:10. Podem ajudar-nos de les guies que hem creat pero això, o bé utilitzar l'inspector de *Propiedades*.

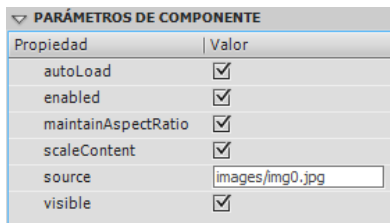


### Pas 4 de 14

En l'inspector de *Propiedades* podem veure que apareix la secció **Parámetros de componente** en la qual podem modificar algunes de les propietats d'aquest component.

Des d'aquí podem indicar un valor per a **source**, per a indicar la ruta del contingut que volem que es mostri en aquest contenidor.

Farem que en obrir l'aplicació es mostri la primera imatge, per la qual cosa en source podem indicar `images/img0.jpg`, ja que és la ruta relativa de la imatge respecte al nostre arxiu Flash.



Provem la pel·lícula amb **Ctrl+Intro** per comprovar que la imatge es carrega correctament.

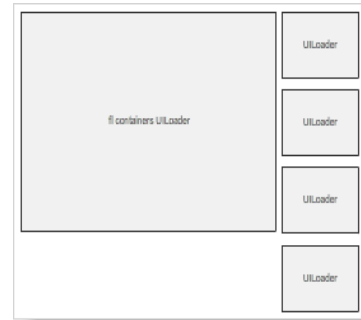
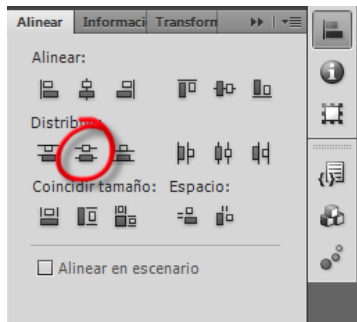
Per a les miniatures utilitzarem també el component UILoader. Creem una altra instància del component i li donem els valors **AN:120** i **AL:90**. Col·loquem aquesta instància en l'extrem superior dret de l'escenari amb ajuda de les guies.

Per crear les altres tres miniatures podem partir d'una còpia d'aquesta última, i així ja tindran els valors d'amplada i altura apropiats

## Pas 5 de 14

Amb ajuda de les guies, col·loquem una altra de les miniatures en la part inferior dreta de l'escenari. Quedarà posicionada en X:420 i Y:330.

Per alinear les dues miniatures restants podem ajudar-nos de les guies per posicionar el marge dret, i del panell **Alinear** per posicionar l'altura, seleccionant les quatre minitutes i **Distribuir verticalment respecte al centro**.



Ara que ja tenim col·locats tots el contenidors en l'escenari, els hi donarem **noms d'instància** a cadascun d'ells des de l'inspector de *Propiedades*.

Anomenarem *main\_ldr* al contenidor principal, i *thumb0\_ldr*, *thumb1\_ldr*, *thumb2\_ldr* i *thumb3\_ldr* a les miniatures.

Anomenem *loaders* a la capa que conté tots el contenidors.

## Pas 6 de 14

Creem una nova capa anomenada *as* per a la programació.

Abans hem comprovat que si escrivim la ruta d'una imatge en el paràmetre *source* del component, la imatge es mostrarà en el contenidor corresponent.

Si definim la propietat *source* d'una instància mitjançant codi, veurem que funciona de la mateixa manera.

Per exemple, si escrivim la línia:

```
thumb0_ldr.source = "images/thumbs/img0.jpg";
```

comprovarem que es carrega la imatge corresponent en el primer contenidor de les miniatures.

Podríem repetir el mateix codi per a les quatre instàncies només canviant els números corresponents, però podria ser poc pràctic fer-ho d'aquesta manera si la nostra galeria tingués moltes imatges. Vegem una manera d'optimitzar el codi.

En primer lloc definirem una variable que emmagatzemi el nombre d'imatges de la nostra galeria, en aquest cas són quatre.

```
var numImages:uint = 4;
```

En altres tutorials quan tractàvem amb nombres utilitzàvem *Number* com a tipus genèric, però per a nombres sencers positius és més correcte utilitzar el tipus més específic *uint* (en el cas de sencers tant positius com negatius es faria servir el tipus *int*).

Crearem un bucle *for* que s'executi quatre vegades (el nombre de les nostres imatges), i que vagi carregant la imatge corresponent en cada miniatura.

```
for (var i:uint=0; i<numImages; i++)
{
    this["thumb"+i+"_ldr"].source = "images/thumbs/img"+i+".jpg";
}
```

## Pas 7 de 14

En tenir la nostra variable `numImages` el valor de 4, podem veure que en la primera línia del codi:

```
for (var i:uint=0; i<numImages; i++)
```

es defineix que el bucle es repetirà 4 vegades, amb el valor per a `i` de 0, 1, 2 i 3.

Si tens dubtes sobre l'ús del bucle `for` pots consultar el [pas 7 del tutorial 10](#).

Continuant amb l'anàlisi del codi, en el fragment:

```
this["thumb"+i+"_ldr"]
```

es substituirà i pels diferents valors que anirà adoptant en les repeticions, creant amb això les cadenes `this["thumb0_ldr"]`, `this["thumb1_ldr"]`, etc.

El `this` buscarà una instància el nom de la qual coincideixi amb la cadena que es troba entre els claudàtors, pas necessari en escriure els noms d'instància de forma dinàmica, la qual cosa equivaldria a haver escrit directament el nom de la instància.

En el fragment:

```
"images/thumbs/img"+i+".jpg"
```

es substituirà igualment la `i` pel seu valor corresponent en el bucle.

Per tant, aquest bucle equivaldrà a haver escrit:

```
thumb0_ldr.source = "images/thumbs/img0.jpg";  
thumb1_ldr.source = "images/thumbs/img1.jpg";  
thumb2_ldr.source = "images/thumbs/img2.jpg";  
thumb3_ldr.source = "images/thumbs/img3.jpg";
```

la qual cosa mostrarà les quatre imatges en el seu corresponent contenidor.

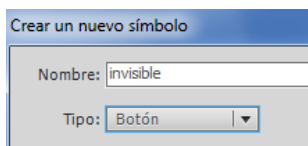
## Pas 8 de 14

El següent que necessitem per a la nostra galeria és que en fer clic a les miniatures es mostri en el contenidor principal la imatge corresponent.

Per a reconèixer amb més facilitat que les miniatures tenen la funció de botons, podem variar lleugerament el seu aspecte al situar-nos o fer clic sobre elles, per exemple il·luminant-les o enfosquin-les segons la nostra interacció.

Una manera eficaç de crear aquesta funcionalitat i canviar l'aspecte de totes les miniatures, és crear un botó invisible i col·locar una instància d'aquest sobre cada miniatura.

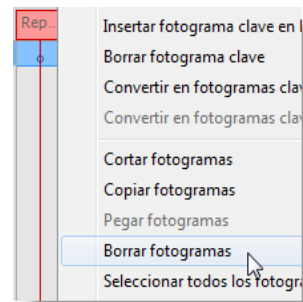
Seleccionem **Insertar > Nuevo símbolo**, tipus **Botón**, i l'anomenem *invisible*.



Dibuixem un rectangle sense traç, amb farciment negre amb un alfa del 50% i amb una mida de 120x90 (el mateix que les miniatures). Alineem el rectangle en la posició 0x0 del seu escenari.

Premem **F6** sobre els fotogrames *sobre i presionado* per crear fotogrames clau amb el mateix contingut que el fotograma *reposo*.

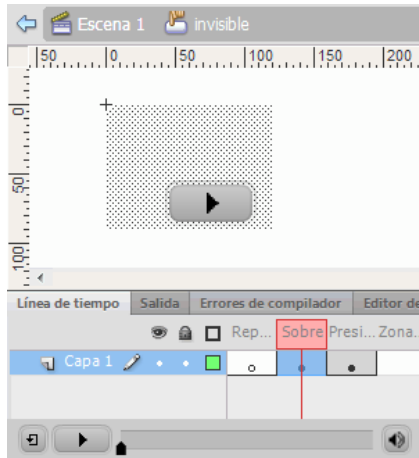
Esborrem el contingut del fotograma *reposo* fent clic amb el **botó dret** del ratolí sobre el fotograma *reposo* i seleccionant **Borrar fotogramas**.



Canviem el color del fotograma *Sobre* de negre a blanc, i amb una transparència del 20%.

## Pas 9 de 14

D'aquesta manera haurem construït un botó que en el seu estat de repòs no es veu (però és igualment funcional). Si som sobre ell mostrarà un requadre blanc semitransparent (semblarà il·luminar el que es trobi sota d'ell) i, de la mateixa manera, enfosquirà el que estigui darrere seu mentre estiguem fent clic sobre d'ell.



Un cop creat el botó, tornem a l'escena principal.

Creem una nova capa anomenada *buttons* per sobre de la capa *loaders*.

Arrosseguem a l'escenari quatre instàncies del botó *invisible* i col·loquem cadascuna d'elles sobre cada contenidor de les miniatures.

Com que el botó *invisible* no té cap gràfic en el seu estat de repòs, veurem un requadre blau per indicar la posició i la mida de la instància del botó en l'escenari, i d'aquesta manera facilitar el seu posicionament.

Encara que aquests botons no tinguin funcionalitat, podem provar la nostra pel·lícula per veure l'efecte d'il·luminar i enfosquir que hem aconseguit amb ells.

Donem a aquests botons els noms de instància *thumb0\_btn*, *thumb1\_btn*, *thumb2\_btn* i *thumb3\_btn*.

El pas següent serà afegir el codi necessari perquè aquests botons que hem creat permetin carregar les imatges corresponents a cada miniatura.

## Pas 10 de 14

En haver donat a cadascun dels botons un nom d'instància amb un nombre que correspon també al de cada imatge, podem afegir la seva funcionalitat en el mateix bucle en el qual hem carregat les imatges de les miniatures.

Afegim en el bucle `for` la línia:

```
this["thumb"+i+"_btn"].addEventListener(MouseEvent.CLICK, loadImage);
```

Amb aquesta línia haurem afegit un listener a cada botó, cridant tots ells a la funció `loadImage`.

Com tots els botons criden a la mateixa funció, aquesta haurà de recuperar el nom del botó que la va llançar per saber quina imatge carregar en el contenidor principal. Com vam veure en altres tutorials, podem identificar el nom de la instància que desencadena l'esdeveniment amb `e.target.name`.

Provem amb la següent definició de la funció `loadImage`:

```
function loadImage(e:MouseEvent):void
{
    trace(e.target.name);
}
```

Si provem la nostra pel·lícula i premem sobre els diferents botons, comprovarem que, efectivament, apareixen els noms de les instàncies en el panell *Salida*.

El nostre contenidor principal (`main_ldr`) haurà de carregar una imatge que es troba en la carpeta `images`, i el nom del qual és `img` seguit del mateix nombre que conté el nom de la instància del botó que hem premut, i seguit de `.jpg` (per exemple `images/img2.jpg`).

En les cadenes de text es considera que el primer caràcter ocupa la posició 0, el segon la posició 1, etc. Amb el mètode `substr` podem extreure caràcters d'una cadena, sent el primer paràmetre la posició des d'on s'extrauran els caràcters i el segon el nombre de caràcters a extreure.

En el nom de les instàncies dels botons, el nombre ocuparia la posició 5. És per això que ens interessa extreure un caràcter que es troba en la posició 5 del nom. Per fer-ho haurem d'escriure `substr(5,1)`.

La nostra funció `loadImage` quedarà per tant com s'indica a continuació:

```
function loadImage(e:MouseEvent):void
{
    main_ldr.source = "images/img"+e.target.name.substr(5,1)+".jpg";
}
```

Provem de nou la nostra pel·lícula per comprovar la càrrega de les diferents imatges segons la miniatura que premem.

## Pas 11 de 14

Finalment inclourem una caixa de text en la nostra minigaleria, que col·loquem en una nova capa anomenada *text*.

Per a aquesta caixa de text utilitzarem un altre component anomenat **TextArea**, que és dins la carpeta **User Interfaces** del panell **Componentes**.

Col·loquem en l'escenari una instància d'aquest component. Li donem unes dimensions de *AN:400* i *AL:100*, i el situem en la part inferior esquerra de l'escenari amb l'ajuda de les guies (*X:10* i *Y:320*).

Podem veure els paràmetres associats a aquest component en l'àrea **Parámetros de Componente** de l'inspector de *Propiedades*.

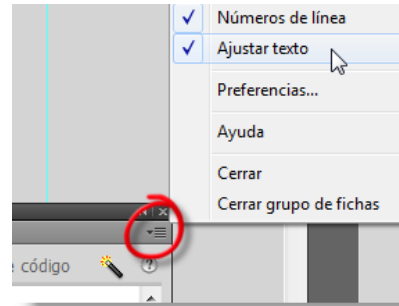
Per conèixer la definició o funcionalitat del diferents paràmetres i altres propietats o mètodes associats a aquest component, podem buscar en l'ajuda de flash la informació sobre la seva classe associada, el nom de la qual apareix en l'àrea de vinculació del panell *Biblioteca*. En aquest cas la classe és [fl.controls.TextArea](#).

A la nostra instància del component *TextArea* l'anomenem *textBox*.

Per afegir text sense format en la nostra caixa de text utilitzarem la propietat *text*. Afegirem la línia següent en el codi:

```
textBox.text="Lorem ipsum dolor sit amet, consectetur adipiscing elit. Donec vitae dui nulla, et gravida justo.";
```

Per poder veure el text amb comoditat mentre programem, sense necessitat d'utilitzar el scroll horitzontal, seleccionem **Ajustar texto** en les opcions del panell *Acciones*.



## Pas 12 de 14

Provem la nostra pel·lícula. Apareixerà el text amb el format predeterminat dins de la caixa de text. No apareixerà cap scroll, ja que en aquest cas no és necessari per l'extensió del text.

Provem a afegir més text a `textBox.text`, per exemple:

```
"Lorem ipsum dolor sit amet, consectetur adipiscing elit. Donec vitae dui nulla, et gravida justo.
```

```
Maecenas quam risus, suscipit ut congue ac, ultricies eget odio. Etiam porta ultrices ante. Phasellus auctor pulvinar auctor. Integer eget vestibulum sem.
```

```
Sed semper tortor vel justo pharetra volutpat. Ut posuere arcu at felis convallis laoreet. In hac habitasse platea dictumst."
```

Si copiem aquest text amb salts de línia inclosos i provem la nostra pel·lícula, el panell **Errores de compilador** mostrarà un error que indica que les cadenes no poden contenir salts de línia.

Podem afegir retorns de carro dins d'un camp de text mitjançant el caràcter de retorn de carro `\r`, substituint aquest caràcter tots els salts de línia que teníem. D'aquesta manera el text quedaria de la següent manera:

```
"Lorem ipsum dolor sit amet, consectetur adipiscing elit. Donec vitae dui nulla, et gravida justo.\r\rMaecenas quam risus, suscipit ut congue ac, ultricies eget odio. Etiam porta ultrices ante. Phasellus auctor pulvinar auctor. Integer eget vestibulum sem.\r\rSed semper tortor vel justo pharetra volutpat. Ut posuere arcu at felis convallis laoreet. In hac habitasse platea dictumst."
```

Provem de nou la pel·lícula i comprovem que l'error anterior ja no apareix. S'haurà generat un scroll vertical de manera automàtica, ja que l'extensió del text és ara més gran que la mida de la caixa.

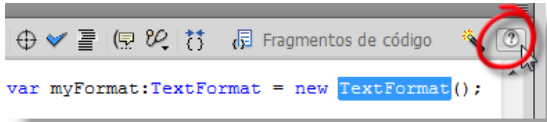
## Pas 13 de 14

Formatarem la nostra caixa de text mitjançant codi. Per fer-ho afegim la línia:

```
var myFormat:TextFormat = new TextFormat();
```

En aquesta línia hem creat una instància de la classe `TextFormat` a la qual hem anomenat `myFormat`.

Per conèixer les propietats de la classe `TextFormat` podem seleccionar el seu nom en el panell *Acciones* i prémer sobre el botó d'ajuda del panell.



Especifiquem la font, el color i la mida que aplicarem al nostre text, per exemple:

```
myFormat.font = "Georgia";  
myFormat.color = "0x003366";  
myFormat.size = 15;
```

Finalment, apliquem aquest format al nostre text:

```
textBox.setStyle("textFormat", myFormat);
```

Amb això ja haurem acabat la nostra minigaleria d'imatges amb un camp de text.

## Pas 14 de 14

Per complementar els conceptes desenvolupats en aquest tutorial, es recomana fer les activitats següents:

1. Mostra en el camp de text diferent contingut segons la imatge carregada.
2. Afegeix un component *Label* per al títol de cada imatge.

