CARICAMENTO DI CONTENUTI ESTERNI

TIPO DI CONTENUTI MULTIMEDIALI

- In Flash posso caricare contenuti multimediali presenti su disco. A secondo dei contenuti utilizzerò oggetti e sistemi di controllo diversi:
  - Filmati Flash
  - Immagini di tipo bitmap
  - File audio
  - Video

TIPO DI CARICAMENTO

- Il caricamento può avvenire secondo tre modalità:
  - **Preloading**: Il contenuto viene completamente caricato prima dell'esecuzione
  - **Progressive downloading**: il contenuto viene eseguito durante il caricamento.
  - **Streaming**: un server dedicato ottimizza la spedizione dei pacchetti in modo che il contenuto multimediale possa essere eseguito durante il caricamento in manierz ottimizzata.

IMMAGINI BITMAP

- Per le immagini Bitmap si usa la modalità preloading.
- Le immagini bitmap vengono caricati in un oggetto di tipo **MovieClip** utilizzando l'oggetto **MovieClipLoader** per controllare il caricamento.
- Sarà necessario segnalare all'utente il progresso del caricamento.
- Flash Player versione 7 è in grado di visualizzare solo immagini in formato jpeg non interallacciate
- Flash Player versione 8 è in grado di visualizzare immagini in formato jpeg interallaciato e non interallacciate, immagini in formato GIF (con eventuale trasparenza) e immagini in formato PNG (con eventuale trasparenza)

FILMATI FLASH

- I filmati flash vengono caricati utilizzando la classe MovieClip.
- Possono essere ottimizzati:
  - Verificare la qualità del sonoro
  - Verificare la qualità delle immagini bitmap
  - Trasformare gli oggetti vettoriali molto complicati in immagini
- Possono essere caricati sia in **progressive downloading** che essere in **preloading**.

FILMATI FLASH: PRELOADING

- Si deve utilizzare la modalità preloading (precaricamento) quando si carica un gioco in cui è necessario caricare l'intero filmato per potere iniziare la partita o un'animazione che non consente un flusso costante di dati in caricamento.
- Il filmato deve essere caricato in un oggetto di tipo **MovieClip** utilizzando l'oggetto **MovieClipLoader** per controllare il caricamento.
- Sarà necessario segnalare all'utente il progresso del caricamento.
- Bisogna tener conto che il filmato verrà eseguito alla velocità impostata per il filmato che carica il contenuto esterno. Le impostazioni del filmato caricato saranno ignorate.

FILMATI FLASH: PROGRESSIVE DOWNLOADING

- La modalità progressive downloading è consigliabile nel caso si carici un'animazione che consente un flusso costante di dati in caricamento.
- Il filmato deve essere caricato in un oggetto di tipo **MovieClip** utilizzando il metodo **MovieClip.LoadMovie**. Appena verrà completato il caricamento del primo frame inizierà l'esecuzione.
- Utilizzando proprietà e metodi della classe MovieClip è possibile fornire all'utente strumenti per il controllo dell'esecuzione del filmato.
- Bisogna tener conto che il filmato verrà eseguito alla velocità impostata per il filmato che carica il contenuto esterno. Le impostazioni del filmato caricato saranno ignorate.

FILE VIDEO

- I video Flash hanno l'estesione **FLV** (**FL**ash **V**ideo)
- Vengono caricati ed eseguiti utilizzando gli Oggetti Video e NetConnection
- Flash 8 offre una componente Video Player personalizzabile
- I video vengono mostrati in modalità streaming se vengono caricati da un server Macromedia, altrimenti vengono eseguiti in progressive downloading
- Calcolo del bit rate:
  - (Bytes * 8) / durata (in secondi).

FILE AUDIO

- Oggetto Sound
- File in formato MP3 cbr (constant bit rate)
- Calcolo del bit rate (vale anche per il video):
  - (Bytes * 8) / durata (in secondi).

CARICAMENTO DI UN FILMATO O DI UN IMMAGINE DA DISCO

- Per caricare dinamicamente un filmato (.swf) o un'immagine jpeg (.jpg) da disco ho a disposizione tre strategie:
  - Caricare il contenuto al posto del filmato principale
  - Caricare il contenuto nel filmato principale su un altro level
  - Utilizzare una MovieClip per caricare il contenuto esterno.
- L'utilizzo di una MovieClip mi dà poi due ulteriori alternative:
  - Utilizzazione del metodo MovieClip.loadMovie
  - Utilizzo dell'oggetto MovieClipLoader.

SOSTITUZIONE DEI CONTENUTI DEL FILMATO PRINCIPALE

- Se semplicemente voglio sostituire il contenuto del mio filmato principale con un altro filmato presente su disco posso usare il comando globale loadMovie() che accetta come parametro l'url del filmato da caricare.
- Devo tener presente che:
  - Tutti i contenuti presenti nello stage
  - Tutti i contenuti presenti nella libreria del filmato
  - Tutte le funzioni e le classi definite nel filmato

  andranno persi.

UTILIZZO DEI LIVELLI

- Se invece voglio aggiungere dei contenuti al filmato principale o sostituirne una parte posso utilizzare i livelli o le movie clip.
- Per utilizzare i livelli userò il comando loadMovieNum(url, livello)
- I contenuti verranno caricati nel livello specificato nel parametro
- Il livello è un'oggetto di tipo MovieClip che viene creato automaticamete (se già non esiste) e che si chiama _level*n*

UTILIZZO DI UNA MOVIECLIP

- Un'alternativa ai livelli (molto più flessibile e da noi preferita) è l'utilizzo di una MovieClip gia presente nel filmato o creata appositamente.
- Basta applicare il metodo loadMovie() alla MovieClip.
- Se la clip è vuota, dopo il caricamento il suo contenuto sarà costituito dal filmato o dall'immagine scaricata
- Se ha del contenuto il contenuto sarà sostituito dal contenuto caricato.

UTILIZZO DI MOVIECLIPLOADER

- MovieClipLoader è una classe presente in flash dalla versione 7 in poi molto utile per il controllo del caricamento di contenuti in un filmato.
- Come per tutte le classi prima di tutto dovrò creare un'istanza della per utilizzarla:

```
var myLoader:MovieClipLoader = new MovieClipLoader();
```

- Dopo di che potrò applicare all'istanza creata il metodo loadClip(url, movieClip) che mi consente di caricare un contenuto esterno in una movie clip

```
myLoader.loadClip("unfilmato.swf", my_clip_mc);
```

UTILIZZO DI MOVIECLIPLOADER

- Il vantaggio rispetto all'uso diretto di MovieClip è ch l'oggetto mi fornische 5 gestori eventi che mi consentono un pieno controllo del caricameto:
  - onLoadStart che mi comunica che il caricamento è iniziato
  - onLoadProgress che mi comunica continuamente lo stato del caricamento
  - onLoadComplete che mi dice che il caricamento è completato
  - onLoadInit che mi comunica che, terminato il caricamento, il filamto è stato inizializzato e quindi i suoi metodi e le sue proprietà sono disponibili
  - onLoadError che mi segnala errori

GESTIONE DEGLI EVENTI NEI CLIP FILMATO

- Un metodo di un gestore di eventi è un metodo di una classe che viene richiamato quando si verifica un evento su un'istanza di tale classe.
- La classe MovieClip, ad esempio, definisce numerosi gestori di eventi che vengono richiamati ogni volta che l'evento in questione si verifica.
- Gestire un evento significa scrivere una funzione anonima che contenga il codice da eseguire quando l'evento si verifica.
- Un gestore di eventi è costituito da tre elementi:
  - l'oggetto al quale si applica l'evento,
  - il nome del metodo del gestore di eventi dell'oggetto
  - la funzione assegnata al gestore di eventi.
- Ecco la struttura di base di un gestore di eventi:

```
object.eventMethod = function () {
// Inserire qui il codice che risponde all'evento.
}
```

UNA SUPERFICIE SU CUI DISEGNARE

CREAZIONE DI CLIP FILMATO IN FASE DI RUNTIME

- Le istanze di clip filmato possono essere realizzate non solo nell'ambiente di creazione di Flash, ma anche in fase di runtime.
- Ogni istanza di clip filmato creata in fase di runtime deve avere:
  - un nome di istanza unico nel MovieClip in cui è contenuto
  - un valore di profondità (*z*-order) ugualmente unico
- La profondità specificata determina in che ordine il nuovo clip si sovrappone agli altri clip sulla stessa linea temporale.
- Per ogni livello di profondità avremo un unico clip. Creando un clip ad un livello già occupato da un altro clip quest'ultimo viene sostituito dal nuovo clip creato.

CREAZIONE DI CLIP FILMATO IN FASE DI RUNTIME

- Posso creare un nuovo MovieClip solo in MovieClip già esitente. Il MovieClip che contiene il MovieClip creato (runtime o design time) viene indicato dalla proprietà **_parent** del MovieClip.
- Ho tre metodi a disposizione per creare un nuovo MovieClip in un MovieClip esistente:
  - **MovieClip.createEmptyMovieClip()**
  - **MovieClip.attachMovie()**
  - **MovieClip.duplicateMovieClip()**

CREAZIONE DI UN CLIP FILMATO VUOTO

- Per creare una nuova istanza di clip filmato vuota sullo stage, utilizzare il metodo createEmptyMovieClip() della classe MovieClip. Questo metodo crea un clip filmato secondario del clip che richiama il metodo.

```
parent_mc.createEmptyMovieClip("new_mc", 10);
```

- Il seguente codice crea un nuovo clip filmato denominato canvas_mc nella linea temporale principale del file SWF in cui viene eseguito lo script e quindi chiama loadMovie() per caricare un file JPEG esterno.

```
this.createEmptyMovieClip("canvas_mc", 10);
canvas_mc.loadMovie("http://www.helpexamples.com/flash/images/image1.jpg");
```

DUPLICAZIONE O RIMOZIONE DI UN CLIP FILMATO

- Per duplicare o rimuovere istanze di clip filmato, utilizzare metodi della classe MovieClip **duplicateMovieClip()** o **removeMovieClip()**.
- Il metodo duplicateMovieClip() crea una nuova istanza da un'istanza di clip filmato esistente e le assegna un nuovo nome di istanza e una profondità, o *z*-order.
- Per eliminare un clip filmato creato con duplicateMovieClip(), utilizzare **removeMovieClip().**

AGGIUNTA DI PARAMETRI A CLIP FILMATO CREATI DINAMICAMENTE

- Se si crea o si duplica dinamicamente un clip filmato utilizzando MovieClip.attachMovie() e MovieClip.duplicateMovie(), è possibile assegnare un valore iniziale ad alcune proprietà.
- Il metodo accetto un parametro opzionale di tipo Object attreaverso il quale posso inizializzare ilclip filmato creati in modo dinamico.

GESTIONE DELLE PROFONDITÀ NEI CLIP FILMATO

- Ogni clip filmato ha un proprio spazio *z*-order che determina in che modo gli oggetti si sovrappongono all'interno del file SWF o del clip filmato di origine. A ciascun clip filmato è associato un valore di profondità che determina se il filmato si trova in primo piano o dietro altri clip filmato della stessa linea temporale.
- Quando si crea un clip filmato in fase di runtime attraverso attachMovie, duplicateMovieClip o createEmptyMovieClip, occorre sempre specificare una profondità per il nuovo clip come parametro del metodo.
- Se si crea o si associa un nuovo clip filmato a una profondità in cui è già presente un altro clip filmato, il nuovo clip o il clip associato sovrascrive il contenuto esistente. Per evitare l'insorgere di questo problema, utilizzare il metodo **MovieClip.getNextHighestDepth**();
- Il valore intero restituito da questo metodo indica la successiva profondità disponibile che determina la posizione in primo piano rispetto a tutti gli altri oggetti del clip filmato.

I METODI PER DISEGNARE

- È possibile disegnare linee e riempimenti sullo stage con Action Script, utilizzando i metodi di disegno della classe MovieClip.
- L'oggetto MovieClip a cui si applicano i metodi è la superficie su cui si disegna. La definizione e la modifica delle proprietà della MovieClip (colorazione, rotazione, posizione, trasparenza, scala, ecc.) influiranno sull'aspetto dei disegni.
- La superfice della MovieClip è trasparente. Disegnale su una MovieClip è come disegnare su un lucido.
- I metodi di disegna che la classe MovieClip definisce sono i seguenti:
  - ***beginFill()***
  - ***beginGradientFill()***
  - ***clear()***
  - ***curveTo()***
  - ***endFill()***
  - ***lineTo()***
  - ***lineStyle()***
  - ***moveTo()***

DISEGNARE UN LINEA

- Questo codice disegna una linea

```
this.createEmptyMovieClip("line_mc", 10);
line_mc.lineStyle(1, 0x000000, 100);
line_mc.moveTo(0, 0);
line_mc.lineTo(200, 100);
line_mc._x = 100;
line_mc._y = 100;
```

- Viene creata la MovieClip
- Vengono impostate le caratteristiche della linea
- Il penello vien spostato alla posizione 0,0
- Viene disegnata la linea
- Viene spostata la movie clip

DISEGNARE UN TRIANGOLO

- Questo codice disegna una triangolo

```
this.createEmptyMovieClip("tri_mc", 1);
with (tri_mc) {
   lineStyle(5, 0xFF00FF, 100);
   moveTo(200, 200);
   lineTo(300, 300);
   lineTo(100, 300);
   lineTo(200, 200);
}
```

- Invece di ripetere tri_mc posso usare il costrutto with

DISEGNARE UN TRIANGOLO

- Scrivere una funzione che disegna un rettangolo

```
this.createEmptyMovieClip("rectangle_mc", 10);
rectangle_mc._x = 100;
rectangle_mc._y = 100;
drawRectangle(rectangle_mc, 240, 180, 0x99FF00, 100);
function drawRectangle(target_mc:MovieClip, boxWidth:Number,
                       boxHeight:Number, fillColor:Number,
                       fillAlpha:Number):Void {
        with (target_mc) {
                beginFill(fillColor, fillAlpha);
                moveTo(0, 0);
                lineTo(boxWidth, 0);
                lineTo(boxWidth, boxHeight);
                lineTo(0, boxHeight);
                lineTo(0, 0);
                endFill();
        }
}
```

CREIAMO UNA CLASSE

FASI DELLA CREAZIONE DI UNA CLASSE

- Le fasi di realizzazione di una classe possono essere così riassunte:
  - ***Progettazione****:* comprende il lavoro da svolgere prima della scrittura del codice. Si tratta di definire il funzionamento del nuovo oggetto che la classe mette disposizione, e quindi stabilire quali proprietà e metodi saranno necessari.
  - ***Scrittura del codice****.* Si passerà quindi alla fase di scrittura del codice. Dovremo realizzare un file di classe secondo le specifiche che abbiamo visto nel capitolo precedente, preoccupandoci soprattutto della corretta sintassi di quanto scriviamo.
  - ***Test e correzione degli errori***. Dovremo poi creare un file FLA di prova in cui testare il corretto funzionamento della classe e correggere gli eventuali errori
  - ***Inserimento in una o più applicazioni***. Una volta testato il funzionamento della classe passeremo al suo utilizzo in alcune applicazioni flash.

PROGETTAZIONE

- Le fasi di progettazione è probabilmente la più importante riduce possibilità di errore, necessità di aggiustamenti. In altri termini ci fa risparmiare tempo.
- Il fase progettuale dobbiamo:
  - Definire l'obbiettivo del nuovo tipo di oggetto che andiamo a realizzare
  - Definire le funzionalità che ci permettono di conseguire il nostro obbiettivo
  - Definire le proprietà che ci servono per modificare il comportamento dell'oggetto
  - Definire i metodi che consentiranno al nostro oggetto di svolgere le suo funzioni

DEFINIZIONE DELL'OBIETTIVO

- Il nostro obbiettivo è creare un semplice oggetto riutilizzabile che carichi un contenuto esterno e lo mostri sullo stage.
- L'oggetto dovrà mostrare una barra che segnali all'utente il progresso di caricamento del contenuto che voglio caricare
- Chiamerò la classe PBarLoader

FUNZIONALITÀ

- Per il nostro PBarLoader definiamo alcune funzionalità base:
  - Dovrò poter definire in che **posizione** collocare limagine o il filmato caricati.
  - La barra che segnala il progresso di caricamento dovrà essere **collocata al centro** dell'area di caricamento.
  - Dovrò poter definire il **colore** della barra che segnala il progresso del filmato

PROPRIETÀ

- Le proprietà sono variabili che definisco all'interno della classe e che contengono le informazioni e i riferimenti agli oggetti utili al funzionamento della classe.
- Le proprietà possono essere pubbliche o private:
  - Le proprietà pubbliche sono quelle utilizzando le quali personalizzo l'istanza delle classe
  - Le proprietà private sono quelle che uso esclusivamente nel codice scritto nella definizione di classe.
- Inoltre per ogni proprieta dovrò stabilire il tipo e definire un valore di default

PROPRIETÀ

- Dovrò creare un MovieClip per caricare il contenuto esterno e una seconda MovieClip su cui disegnare la barra di caricamento.
- Una MovieClip deve essere creata all'interno di una MovieClip che la contiene.
- Ecco quindi le tre prime proprietà che definiamo

Nome	Descrizione	Visibilità	Tipo	Default
image_mc	MovieClip che verrà usata per caricare il contenuto esterno	private	MovieClip	
progress_mc	MovieClip su cui sarà disegnata la barra di caricamento	private	MovieClip	
parent_mc	MovieClip che contiene image_mc	private	MovieClip	_root

PROPRIETÀ

- Per caricare il contenuto esterno utilizzerò l'oggetto MovieClipLoader.
- Creerò l'oggetto alla creazione del mio **PBarLoader** e memorizzero il suo riferimento in una proprietà. Creerò anche le proprietà onLoadInit, onLoadProgress e onLoadError che conterranno il codice che verrà eseguito in risposta agli eventi dello stesso nome notificati da myLoader:

Nome	Descrizione	Visibilità	Tipo	Default
myLoader	Oggetto MovieClipLoader che utilizzerò per il caricamento dei dati esterni	private	MovieClipLoader	
onLoadInit	Risposta all'evento onLoadInit	private	Function	
onLoadProgress	Risposta all'evento onLoadProgress	private	Function	
onLoadError	Risposta all'evento onLoadError	private	Function	

PROPRIETÀ

- Dovrò comunicare al mio oggetto la posizione, le dimensioni del contenuto da caricare in modo da poter collocare la barra di caricamento al centro.
- Per semplificare per il momento stabilisco che queste proprietà potranno essere impostate solo alla creazione le definisco, quindi, private.

Nome	Descrizione	Visibilità	Tipo	Default
x	Posizione x del contenuto caricato	private	Number	0
y	Posizione y del contenuto caricato	private	Number	0
width	Larghezza del contenuto caricato	private	Number	Stage
height	Altezza del contenuto caricato	private	Number	Stage

PROPRIETÀ

- Infine definirò le proprietà che mi consentono di personalizzare la barra di caricamento: dimensioni e colore.

Nome	Descrizione	Visibilità	Tipo	Default
bgcolor	Colore di sfondo della barra	public	Number	**0xCCCCCC**
color	Colore della barra	public	Number	**0x000000**
bar_width	Larghezza della barra	public	Number	200
bar_height	Altezza della barra	public	Number	5

I METODI

- I metodi sono funzioni che definisco all'interno della classe e che svolgono le operazioni necessarie al funzionamento della classe.
- I metodi, come le proprietà, possono essere pubblici o privati:
  - I metodi pubblici sono quelli che utilizzo per far svolgere agli oggetti istanze della classe la loro funzione
  - I metodi privati sono quelli usati esclusivamente da altri metodi della classe.

I METODI

- In PBarLoader definirò quattro metodi
  - Il constructor della classe che che creerà gli oggetti necessari e inizializzerà le proprietà della classe
  - Il metodo pubblico loadClip (il comando che caricherà il contenuto esterno)
  - Il due metodi privati initBar e writeBar che rispettivamente inizializzaranno e aggiorneranno la barra di caricamento.

I METODI: CONSTRUCTOR

- In constructor è sempre un metodo pubblico che ha lo stesso nome della classe. Potrà avere da uno a cinque parametri:
  - ***PBarLoader(Contenitore:MovieClip, xpos:Number, ypos:Number, largehzza:Number, altezza:Number)***.
- Il primo parametro è obbligatorio, gli altri mi consentono di inizializzare delle proprietà per cui ho previsto in alternativa dei valori di default. Azioni del metodo:
  - Inizializzazione della proprietà **parent_mc**.
  - Creazione della MovieClip ***image_mc***
  - Creazione della MovieClip ***progress_mc***
  - Creazione dell'oggetto MovieClipLoader ***myLoader***
  - Inizializzazione delle proprietà ***x***, ***y***, ***width*** e ***height***.
  - Registrazione del PBarLoader creato in myLoader affinchè questo notifichi gli eventi legati al caricamento

I METODI: CARICAMENTO

- Per caricare il contenuto esterno costruirò un metodo con un unico parametro il nome del file da caricare:
  - ***loadClip(url: String)***.
- Il metodo:
  - Inizializzerà la barra di caricamento.
  - Utilizzerà myLoader per caricare il file

I METODI: LA BARRA

- Disegna lo sfondo della barra secondo il colore e le dimensioni definite e la posiziona al centro :
  - ***initBar()***.
- Disegna la barra nella dimensione che rappresenta lo stato del caricamento (il parametro rapresnta il rapporto tra byte scaricati e bytes totali:
  - ***writeBar(caricamento:Number)***;