ANCORA SULLE CLASSI DERIVATE DALLA CLASSE MOVIECLIP

LA CLASSE MOVIECLIP

- La classe MovieClip è l'oggetto software che ci consente di gestire gli elementi visuali in Flash da ActionScript.
- Attraverso la classe Movie Clip posso gestire:
  - MovieClip creati runtime utilizzando ActionScript.
  - Simboli di tipo MovieClip memorizzati in libreria e trascinati durante lo sviluppo sullo stage.
  - Simboli di tipo MovieClip memorizzati in libreria e caricati sullo stage in fase runtime.
  - Filmati memorizzati su disco e caricati runtime all'interno del filmato principale.
  - Immagini bitmap (formato jpeg) memorizzate su disco.

LA CLASSE MOVIECLIP

- La classe MovieClip normalmente NON si crea utilizzando il comando **new**.
- Quando trascino un Simbolo di tipo MovieClip memorizzato in libreria sullo stage durante lo sviluppo automaticamente creo un oggetto MovieClip che è istanza del simbolo in libreria.
- Posso creare MovieClip runtime all'interno di altre MovieClip applicando il metodo **CreateEmptyMovieClip().**
- Posso inserire istanze di simboli di tipo MovieClip memorizzati in libreria in altre MovieClip applicando il metodo **AttachMovieClip().**
- Posso caricare filmati o immagini bitmap memorizzati su disco sostituendo il contenute di MovieClip esistenti o creandone di nuove.

MOVIECLIP CREATA DESIGN TIME

- Quando trascino un simbolo MovieClip sullo stage creo un nuova istanza di quel simbolo.
- Se a quella istanza design time do un nome, attraverso quel nome posso modificare le proprietà di quella istanza e applicare ad essa i metodi di MovieClip.
- Intermini di Programmazioone ad Oggetti quando creo un simbolo MovieClip creo un modello di filmato che condivide con la classe MovieClip tutte le proprietà e tutti i metodi, e che aggiunge alla MovieClip astratta i contenuti visuali presenti.
- L'istanza creata appartiene alla MovieClip (film principale o secondario) entro alla quale è stata inserita. Programmaticamente la si puo raggiungere utilizzando indifferentemente la sintassi degli Oggetti **contenitore.nomeistanza** che la sintassi degli array associativi **contenitore["nomeistanza"]**

MODIFICA DELLA POSIZIONE E DELL'ASPETTO DI UN CLIP FILMATO

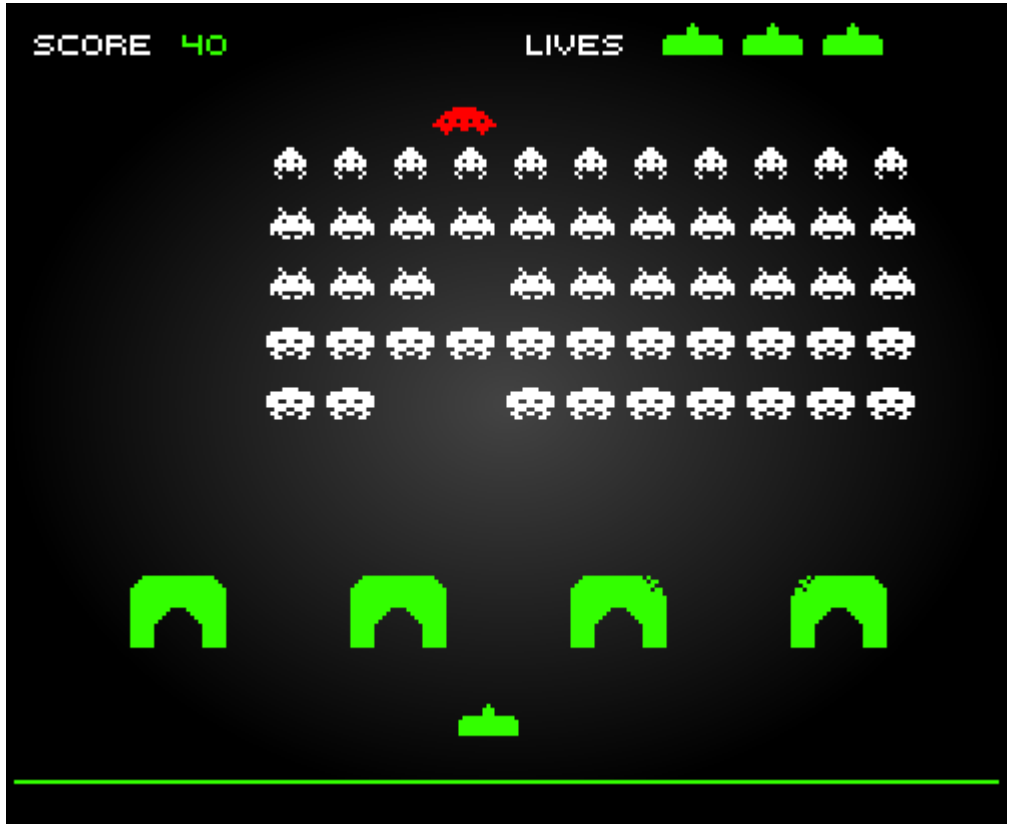
- Per modificare le proprietà di un clip filmato durante l'esecuzione, creare un'istruzione che assegna un valore a una proprietà.
- Le proprietà _x, _y, _rotation, _xscale, _yscale, _height, _width, _alpha e _visible determinao l’aspetto geometrico del clip filmato di tutti gli elementi in esso contenuto.
- Le seguenti proprità sono a sola lettura: _currentframe, _droptarget, _framesloaded, _parent, _target, _totalframes, _url, _xmouse e _ymouse.

CREAZIONE DI UNO SPRITE

CHE COS'È UNO SPRITE

- lo **sprite** è una figura bidimensionale, che può essere spostata in un campo di gioco indipendentemente dagli altri oggetti grafici presenti (figure di sfondo, altri sprite, ecc.) seguendo determinate regole di comportamento.
- Normalmento un sprite può reagire a due tipi di eventi:
  - Comandi dell'utente (quando lo sprite è controllato direttamente dall'utente – goicatore)
  - Collisione con altri spite.

UN ESEMPIO:



PROGRAMMARE UNO SPRITE IN FLASH

- La classe MovieClip è lo strumento che abbiamo per realizzare uno sprite
- La classe MovieClip ha già molte delle caratteristiche di un sprite. Quelle che mancano le possiamo aggiungere noi con la programmazione.
- La maniere più semplice e logica pre creare uno sprite è quindi creare una classe derivata dalla classe MovieClip.

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.

DEFINIZIONE DELL'OBIETTIVO

- Per sperimentare la creazioni di sprite proveremo a realizzare un’applicazione dove:
  - Verranno creati oggetti rotanti che si muoveranno casualmente nello stage.
  - Quando questi oggetti colliderano con altri oggetti, esploderanno e scompariranno.

DEFINIZIONE DELL'OBIETTIVO

- Per gestire questi oggetti creeremo una classe che chiameremo RandomSprite.
- Questa classe sarà un “estensione” della classe MovieClip. Ne ereditareà. Cioè tutte le proprietà e i metodi.
- Alle proprietà e ai metodi della classe MovieClip aggiungeremo le proprietà e i metodi specifici per gestire le funzionalità della classe RandomSpite

FUNZIONALITÀ

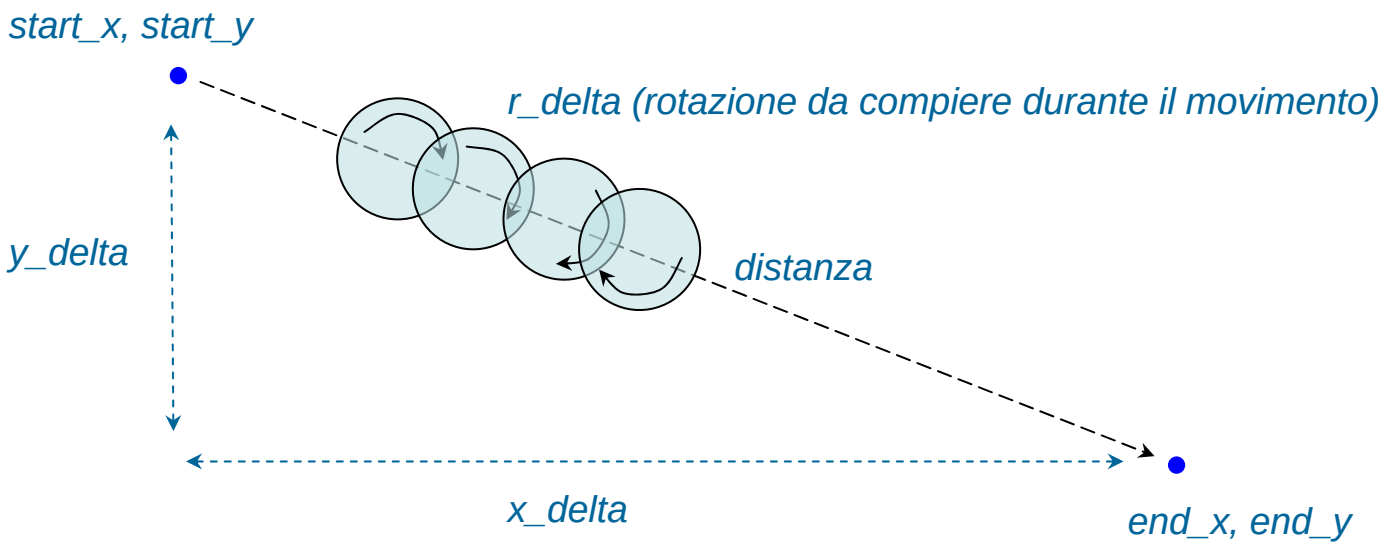
- Per la nostra classe RandomSprite definiamo le funzionalità base:
  - **Gestione del movimento**. (RomdomSprite avrà un metodo che le consentirà di muoversi da un ponto all'altro dello schermo).
  - **Velocità** del movimento (il movimento dovrà avvenire ad una velocità determinata).
  - Durante il movimento RandomSprite **rotolerà** verso la destinazione.
  - **Rilevamento delle collisioni** (RandomSprite rileverà se entra in collisione con altri sprite).
  - Dovrà essere definita un metodo che faccia **esplodere** RandomSprite nel caso fosse necessario.
  - Infine andrà risolto il problema di **creare** gli sprite che devono apparire sullo schermo e di **eliminare** quelli esplosi per liberare risorse.

FUNZIONALITÀ

- Affronteremo la creazione della nostra applicazione articolandola in quattro punti:
  - **Animazione del movimento**.
  - **Rilevamento delle collisioni.**
  - **Esplosione.**
  - **Creazione dinamica degli sprite da fare apparire sullo schermo**.
- I primi tre punti saranno gestiti attraverso metodi della classe RandomSprite, l'ultimo dall'applicazione flash che utilizza la classe. La distruzione (e relativa liberazione di risorse) degli sprite inutilizzati sarà gestita dalla classe RandomSprite contestualmente all'evento collisione/esplosione.

GESTIONE DEL MOVIMENTO

```
x_delta = end_x - start_x;
y_delta = end_y - start_y;
distanza = Math.sqrt((x_delta * x_delta) + (y_delta * y_delta))
totalTime = distanza/speed * 1000
r_delta = distanza/circonferenza * 360;
```



GESTIONE DEL MOVIMENTO (ALGORITMO)

1. Memorizzo il punto di partenza (**start_x** e **start_y, start_rotation**)
2. Memorizzo il tempo di partenza
3. Calcolo il tempo totale
4. Creo un evento **onEnterFrame** che:
    1. Controlla quanto tempo è trascorso
    2. In proporzione al tempo trascorso modifica le proprietà **_x** e **_y** della MovieClip.
    3. In proporzione al tempo trascorso modifica la proprietà **_rotation**.
    4. Controlla se è avvenuta una collisione.
    5. Quando il tempo totale è trascorso interrompe l'evento **onEnterFrame**

RILEVAMENTO DELLE COLLISIONI

- Utilizzerò il metodo **hitTest** della classe MovieClip:

  MovieClip.**hitTest**(target_mc):Boolean;
- Ritorna vero se i perimetri di MovieClip e target_mc si intersecano.
- Dovro utilizzare un ciclo **for…in** per controllare la movie clip in oggetto non collide con nessuna delle altre presenti sullo stage

ESPLOSIONE

- Sarà gestita con una semplice animazione della clip che un apposito metodo farà partire.
- Il metodo si occuperà anche di distruggere la clip alla fina dell'animazione.

CREAZIONE DINAMICA DEGLI SPRITE

- Per creare dinameicamente gli sprite sullo stage si utilizzerò il metodo della classe MovieClip AttachMovie..

ATTACH MOVIE

- Il metodo attachMovie() associa allo stage un'istanza di un simbolo di tipo clip filmato nella libreria. Il nuovo clip viene inserito nel MovieClip a cui è stato applicato il metodo.
- Per poterlo utilizzare da ActionScript il simbolo deve essere esportatp nel file swf.
- Per eseguire questa operazione, utilizzare la finestra di dialogo Proprietà di concatenamento.
- Per impostazione predefinita, tutti i clip filmato esportati per l'utilizzo in ActionScript vengono caricati prima del fotogramma iniziale del file SWF che li contiene.

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 PRIVATE

- Memorizzerò in alcune proprietà private le informazioni che devo consultare durante l'animazione.

Nome	Descrizione	Visibilità	Tipo	Default
start_x e **start_y**	Memorizzano le coordinate da cui è partito il movimento	private	Number	
start_rotation	Memorizza la rotazione della MovieClip al momento in cui è partito il movimento	private	Number	
x_delta e **y_delta**	Di quanto deve spostarsi la MovieClip sugli assi x e y per arrivare a destinazione.	private	Number	
r_delta	Di quanto deve ruotare la MovieClip durante il movimento.	private	Number	
startTime	Tempo segnato dal timer all'inizio del movimento	private	Number	

PROPRIETÀ PUBBLICHE

- Avrò poi due proprietà pubbliche che mi consentono di personalizzare il comportamento della MovieClip: il livello della velocità con cui la MovieClip si sposta e se la MovieClip rotola op scivola.

Nome	Descrizione	Visibilità	Tipo	Default
speed	Determina la velocità in pixel per secondo con cui si muove la MovieClip.	public	Number	100
spriteType	Mi garantisce che la collisione è avvenuta con un certo tipo di sprite. Sarà una proprietà a sola lettura.	public	String	randomSprite (a sola lettura)

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 RandomSprite definirò i seguenti metodi:
  - **private function animMoveTo**(**end_x**:Number, **end_y**:Number) ha il compito di gestire il movimento della movie clip
  - **private function goRandom**() calcola due coordinate casuali nell'area dello stage e fa partire la clip verso la destinazione casuale utilizzando **animMoveTo**.
  - **private function checkCollision**() controlla se la clip è in collisione con un'altra clip e in tal caso lancia explode.
  - **public function explode**() che gestiche l'esplosione e la rimozione della clip.

I METODI: CONSTRUCTOR

- In **constructor** è sempre un metodo pubblico che ha lo stesso nome della classe.
- Nel caso di RandomSprite utilizzerò il constructor e per iniziare la generazione di coordinate casuali che farà vagare il mio sprite sullo stage in maniera casuale.
- Il costructor quindi sarà composto da un solo comando: la chiamata a goRandom.

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().**

ATTACH MOVIE

- Un ultimo modo per creare istanze di clip filmato in fase di runtime consiste nell'utilizzare il metodo **attachMovie()**. Il metodo attachMovie() associa allo stage un'istanza di un simbolo di tipo clip filmato nella libreria. Il nuovo clip viene inserito nel MovieClip a cui è stato applicato il metodo.
- Per poterlo utilizzare da ActionScript il simbolo deve essere esportatp nel file swf.
- Per eseguire questa operazione, utilizzare la finestra di dialogo Proprietà di concatenamento.
- Per impostazione predefinita, tutti i clip filmato esportati per l'utilizzo in ActionScript vengono caricati prima del fotogramma iniziale del file SWF che li contiene.

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.