

LA CLASSE DATE

DATE

- La classe *Date* consente di recuperare i valori relativi alla data e all'ora del tempo universale (UTC) o del sistema operativo su cui è in esecuzione Flash Player.
- Per chiamare i metodi della classe *Date*, è prima necessario creare un oggetto *Date* mediante la funzione di costruzione della classe *Date*.

CREARE UN'ISTANZA DI DATE

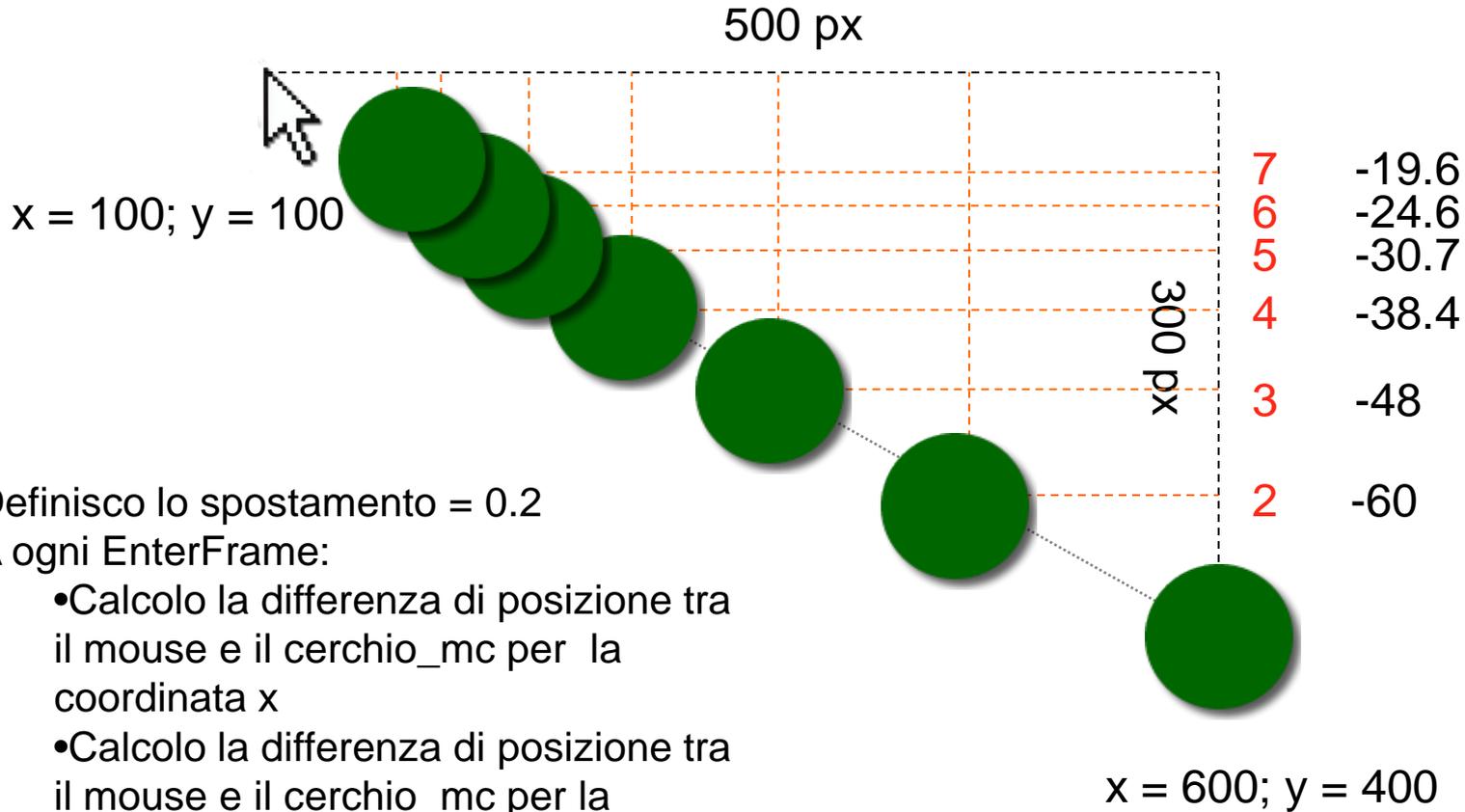
- Posso passare alla funzione una data del passato o del futuro. In questo caso `Date()` richiede da due a sette parametri (anno, mese, giorno, ora, minuti, secondi, millisecondi).
- In alternativa, posso costruire un oggetto *Date* passando un singolo parametro che rappresenta il numero di millisecondi trascorsi dal 1 gennaio 1970 alle 0:00:00 GMT.
- Se, infine, non specifico alcun parametro all'oggetto data `Date()` vengono assegnate la data e l'ora correnti.

USO DI DATE

- Una volta creato all'oggetto Date posso applicare varie proprietà che:
 - Mi restituiscono informazione sull'anno, il mese, il giorno, il giorno della settimana, ecc della data creata.
 - Mi consentono di confrontare, sommare e sottrarre date

CREAZIONE DI CLASSI

Spostamento: **0.2** della distanza



- Definisco lo spostamento = 0.2
- A ogni EnterFrame:
 - Calcolo la differenza di posizione tra il mouse e il cerchio_mc per la coordinata x
 - Calcolo la differenza di posizione tra il mouse e il cerchio_mc per la coordinata y
 - Aggiorno la posizione di cerchio aggiungendo un quinto della distanza calcolato alle sue coordinate

SEGUIMI

```
import flash.display.*;
import flash.events.*;

function aggiornaPosizione (e:Event) {
    cerchio_mc.x = cerchio_mc.x +
        (cerchio_mc.stage.mouseX -
         cerchio_mc.x) * 0.2;

    cerchio_mc.y = cerchio_mc.y +
        (cerchio_mc.stage.mouseY -
         cerchio_mc.y) * 0.2;
}

cerchio_mc.addEventListener(Event.ENTER_FRAME,
                             aggiornaPosizione);
```

LA CLASSE SEGUIMI

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

public class Seguimi {
    private var dObject:DisplayObject;
    private var accelerazione:Number;
    private var contenitore:DisplayObjectContainer;

    public function Seguimi (obj:DisplayObject,e:Number) {
        ease = e;
        dObject = obj;
        contesto = dObject.parent;
    }

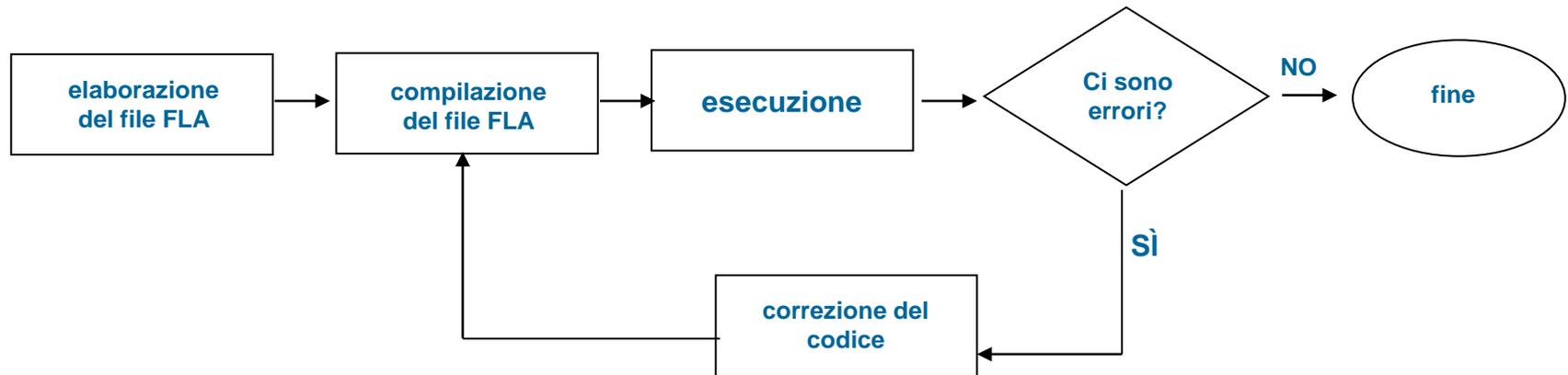
    private function objPos (e:Event) {
        dObject.x = dObject.x + (contesto.mouseX - dObject.x) * ease;
        dObject.y = dObject.y + (contesto.mouseY - dObject.y) * ease;
    }

    public function startFollow () {
        dObject.addEventListener(Event.ENTER_FRAME, objPos);
    }

    public function stopFollow () {
        dObject.removeEventListener(Event.ENTER_FRAME, objPos);
    }
}
}
```

IL PROCESSO

- Come abbiamo già detto il processo di realizzazione di un progetto flash è strutturata in un flusso simile a questo:



- Con la programmazione con le classi introduciamo un nuovo tipo di file: il file **Action Script (.as)** ma il processo rimane lo stesso.
- Se il progetto comprende anche dei file .as il compilatore li unirà ai dati contenuti nel .fla e creerà un unico file shockwave (filmato flash compilato e compresso).

I FILE DI CLASSE

- Il file `.as` è un normale file di testo (tipo blocco note) che contiene codice Action Script.
- Una classe in *ActionScript* viene sempre definita in un file esterno (un normale file di testo con estensione `.as`) che ha lo stesso nome della classe e che viene chiamato *file di classe*.
- Quando un filmato flash viene compilato (utilizzando Controllo > Prova filmato o File > Pubblica) per generare il file `.swf`, il codice contenuto nei file di classe necessari viene compilato e aggiunto al file `.swf`.

I FILE DI CLASSE

- Quando il compilatore trova che nel filmato da compilare viene utilizzata una classe **DEVE** trovare il file che contiene il codice relativo a quella classe:
- Il file di classe deve avere esattamente lo stesso nome della classe (case sensitive).
- Il compilatore deve sapere in che cartelle cercare.

I FILE DI CLASSE

1. Il compilatore in primo luogo inizierà la sua ricerca dalla cartella in cui è stato salvato il file .fla.
2. Esiste un elenco globale di cartelle che contengono classi che si può modificare andando in: Modifica>Preferenze>ActionScript e scegliendo il bottone “Impostazioni Action Script 3”

I PACCHETTI

- In ActionScript 3 è fondamentale capire il concetto di package.
- Con l'aggiunta del comando **package** (obbligatorio) per ogni classe viene definito un **package** di appartenenza che corrisponde a come i file di classe sono organizzati su disco
- Supponiamo che il mio progetto Seguimi fla si collocato in
... **\Esempi\Seguimi**

I PACCHETTI

```
//package aninimo
package {
  public class Seguimi{
    ...
  }
}
```

- Il file di classe si chiamerà **Seguimi.as** e risiederà nella cartella del progetto:

..\Esempi\Seguimi\Seguimi.as

I PACCHETTI

```
//package con nome
package classiProgetto{
    public class Seguimi{
        ...
    }
}
```

Il file di classe si chiamerà **Seguimi.as** e risiederà nella cartella **classiProgetto**:

..\esempi\seguimi\classiProgetto\Seguimi.as

I PACCHETTI

```
//package con nome
package hype.classiPersonali.comportamenti{
    public class Seguimi{
        ...
    }
}
```

Il file di classe si chiamerà **Seguimi.as** e risiederà nella cartella generale in cui ho messo le mie classi secondo questo percorso:

C:\leMieClassi\hype\classiPersonali\comportamenti

I PACKAGES STANDARD DI FLASH CS3/CS4

I PACKAGES STANDARD DI FLASH CS3

- Vedi **Help di Flash**:
 - Guida di riferimento al Linguaggio e alla Componenti di ActionScript 3.0 -> Tutti i packages
- Oppure **Programmare con ActionScript 3.0**
 - Pagina 56

IL FILE DI CLASSE

- Per definire una classe devo creare un file **actionScript** esterno.
- Prima di tutto dovrò definire il package a cui appartiene la classe
- Se il file di classe risiede nella stessa cartella il cui risiede il progetto userò un package anonimo
- Altrimenti specificherò un package che ricalca il percorso in cui il file è memorizzato

```
package {
    public class Seguimi{
        ...
    }
}
```

IL FILE DI CLASSE

- Dovrò poi inserire i comandi import necessari.
- I comandi import dovranno essere definiti **prima** della definizione di classe
- Dovro definire i comandi import sia pe le classi flash che per quelle da me definite

LA CLASSE SEGUIMI

```
package {  
    import flash.display.*;  
    import flash.events.*;  
    public class Seguimi {  
        //definizione della classe  
        ...  
        ...  
        ...  
    }  
}
```

IL FILE DI CLASSE

- Inserirò quindi le dichiarazioni delle proprietà della classe
- Le proprietà di una classe sono variabili e vanno dichiarate nello stesso modo
- La dichiarazione è preceduta da un attributo che ne regola il **controllo di accesso** (scope)

LA CLASSE SEGUIMI

```
package {  
    .....  
    public class Seguimi {  
        // Proprietà della classe  
        private var oggetto: DisplayObject;  
        private var accelerazione: Number;  
    }  
}
```

IL FILE DI CLASSE

- Inserirò quindi il costruttore
- Il costruttore è una funzione con attributo **public** che ha lo stesso nome della classe, viene eseguito quando viene creata un'istanza della classe e quindi deve contenere le istruzioni di inizializzazione

LA CLASSE SEGUIMI

```
public class Seguimi {  
    ..  
    /* Metodo "Constructor" : viene chiamato  
    automaticamente quando si crea una istanza della  
    classe */  
        public function Seguimi (obj:DisplayObject,  
                                   a:Number) {  
            accelerazione = e;  
            oggetto = obj;  
        }  
    ..  
}
```

FILE DI CLASSE

- Scriverò i metodi, raggruppandoli per funzionalità.
- Questo tipo di organizzazione dei metodi consente di migliorare la leggibilità e la chiarezza del codice.

LA CLASSE SEGUIMI

```
public class Seguimi {
    ..
    // Metodo che aggiorna la posizione
    // dell'oggetto
    private function esegui(e:Event) {
        oggetto.x = oggetto.x + (oggetto.stage.mouseX
            - oggetto.x ) * ease;
        oggetto.y = oggetto.y + (oggetto.stage.mouseY
            - oggetto.y) * ease;
    }
}
```

LA CLASSE SEGUIMI

```
public class Seguimi {
    ..
    // Metodi che iniziano e terminano l'animazione
    // dell'oggetto
    public function startSeguimi () {
        oggetto.addEventListener(Event.ENTER_FRAME,
                                objPos);
    }

    public function stopSeguimi () {
        oggetto.removeEventListener(Event.ENTER_FRAME,
                                    objPos);
    }
}
```

PROVARE UNA CLASSE

- Per creare e usare una classe è necessario:
 - Definizione di una classe in un file di classe *ActionScript* esterno.
 - Salvataggio del file di classe nella directory specificata per il percorso della classe (o nel percorso in cui Flash cerca le classi) oppure nella stessa directory del file FLA dell'applicazione.
 - Creazione di un'istanza della classe in un altro script, ossia un documento FLA o un file di script esterno, oppure tramite creazione di una sottoclasse basata sulla classe originale.

USARE UNA CLASSE

- Per creare un'istanza di una classe *ActionScript*, si utilizza l'operatore **new** per richiamare la funzione di costruzione della classe. Tale funzione ha sempre lo stesso nome della classe e restituisce un'istanza della classe che generalmente viene assegnata a una variabile.

```
var s:Seguimi = new Seguimi(miaClip);
```

- Usando l'operatore punto (.) si accede al valore di una proprietà o a un metodo di un'istanza.

```
s.startSeguimi();
```

GLI ATTRIBUTI DI CONTROLLO DI ACCESSO

- Gli attributi di controllo di accesso determinano la visibilità o scopo di classi, proprietà e metodi.
- La sintassi è la seguente
- <attributo> **class**
- <attributo> **var**
- <attributo> **function**

GLI ATTRIBUTI DI CONTROLLO DI ACCESSO

public	Visibilità completa
internal	Visibilità limitata alle classi che si trovano nello stesso package
private	Visibilità limitata alla sola classe di appartenenza
protected	Visibilità limitata alla classe di appartenenza e alle sottoclassi

METODI E PROPRIETÀ STATICI

- La parola chiave *static* specifica che una variabile o una funzione viene creata solo una volta per ogni classe anziché in ogni oggetto basato sulla classe. È possibile accedere a un membro di classe statico senza creare un'istanza della classe. I metodi e le proprietà statici possono essere sia pubblici che privati.
- In ActionScript ci sono classi predefinite che hanno solo metodi e proprietà statiche.