OROLOGIO GENERICO

INGEGNERIZZARE UN PROBLEMA

MOTORE

(Orologio generico)

Aggiornamento periodico dell'ora ricavandola dall'orologio del computer

VISUALIZZAZIONE

(Orologio digitale)

(Orologio analogico)

VISUALIZZAZIONE

(Orologio digitale)

(Orologio analogico)

- Inizializzazione
- Aggiornamento dell’ora

OROLOGIO GENERICO

- Creo una classe astratta da cui derivare orologi specifici
- Compito della classe sarà semplicemente tenere aggiornate (una volta al secondo) tre proprietà che conterranno rispettivamente ora, minuti e secondi.
- Useremo una nuova classe la classe Timer

LA CLASSE TIMER

- La classe timer è una classe fornita con ActionScript 3 che genera un evento a intervalli di tempo prestabilito
- E’ utile quando ho bisogno di generare eventi ripetuti nel tempo indipendenti dal ritmo scandito dai frame
- In questo caso basta aggiornare l’orologio una volta la secondo.

OROLOGIO GENERICO

- Importo solo leclassi che mi consentono di defini lo sprite e il timer.

```
package {
   import flash.display.Sprite;
   import flash.utils.Timer;
   import flash.events.TimerEvent;

   .......
}
```

OROLOGIO GENERICO

- Definisco le proprietà che il mio timer dovrà aggiornare

```
package {
  public class OrologioGenerico extends Sprite {
    protected var ore:uint;
    protected var minuti:uint;
    protected var secondi:uint;

  .......
}
```

OROLOGIO GENERICO

- Come **costructor** definisco una funzione che chiama i metodi necessari a disegnare l'orologio e a inserirvi una valore iniziale:

```
package {
  public class OrologioGenerico extends Sprite {
    public function OrologioGenerico () {
      leggiOra();
      inizializzaVisualizzazione();
      visualizzaOra();
      inizializzaTimer();
   }
  .......
}
```

OROLOGIO GENERICO

- Definisco i metodi che inizializzano il timer e lo fanno partire: **leggiOra** aggiorna le varibili sulla base dell'ora fornita dal computer:

```
package {
  public class OrologioGenerico extends Sprite {
  .......
    protected function leggiOra() {
      var adesso:Date = new Date();
      ore = adesso.getHours();
      minuti = adesso.getMinutes();
      secondi = adesso.getSeconds();
    }
  .......
}
```

OROLOGIO GENERICO

- **inizializzaTimer** e **aggiorna** sono rispettivamente il metodo che crea e fa partire il timer e il metodo che viene chiamato ad ogni evento generato dal timer:

```
package {
  public class OrologioGenerico extends Sprite {
  .......
    protected function inizializzaTimer() {
        var myTimer:Timer = new Timer(1000);
        myTimer.addEventListener(TimerEvent.TIMER, aggiorna);
        myTimer.start();
    }

    private function aggiorna(e:TimerEvent) {
       leggiOra();
       visualizzaOra();
    }
}
```

OROLOGIO GENERICO

- Dichiaro i metodi **inizializzaVisualizzazione** e **visualizzaOra** che lascio vuoti in quanto saranno implementati nelle sub classi derivata da OrologioGenerico.

```
package {
  public class OrologioGenerico extends Sprite {
  .......
    public function inizializzaVisualizzazione() {
        //metodo da definire nelle classi derivate
    }

    public function visualizzaOra () {
        //metodo da definire nelle classi derivate
    }
}
```

OROLOGIO ANALOGICO

HO IMPORTATO LE CLASSI NECESSARIA

- In rosso la classi che devo ancora aggiungere (mi servono per disegnare il quadrante)

```
import flash.display.Sprite;
import flash.display.Shape;
import flash.display.Graphics;
import flash.utils.Timer;
import flash.events.TimerEvent;

import flash.text.TextField;
import flash.text.TextFieldAutoSize;
```

DICHIARAZIONE DELLA CLASSE

1. Dichiaro la classe OrologioAnalogico facendola discendere da Sprite :

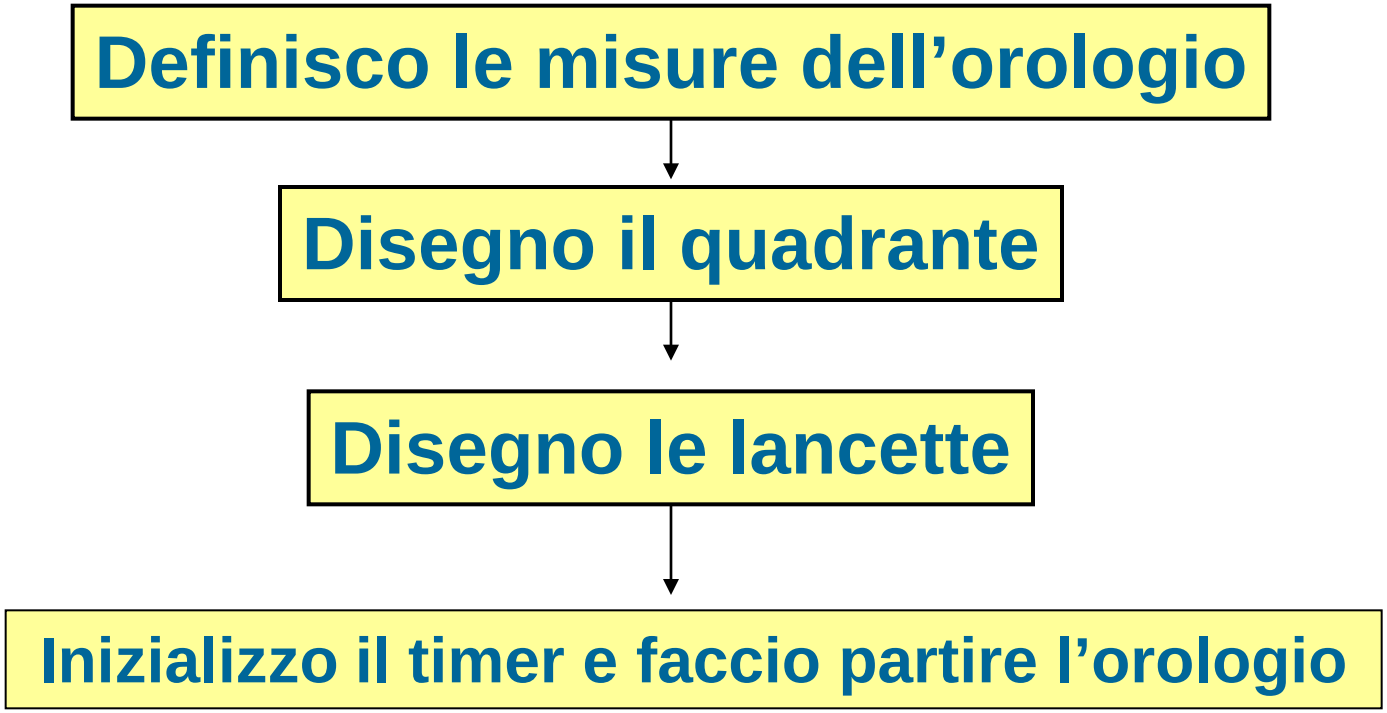
```
package {
  import flash.display.Sprite;
  ......
  public class OrologioAnalogico extends Sprite {
        ......
  }
}
```

HO DICHIARATO DELLE PROPRIETÀ

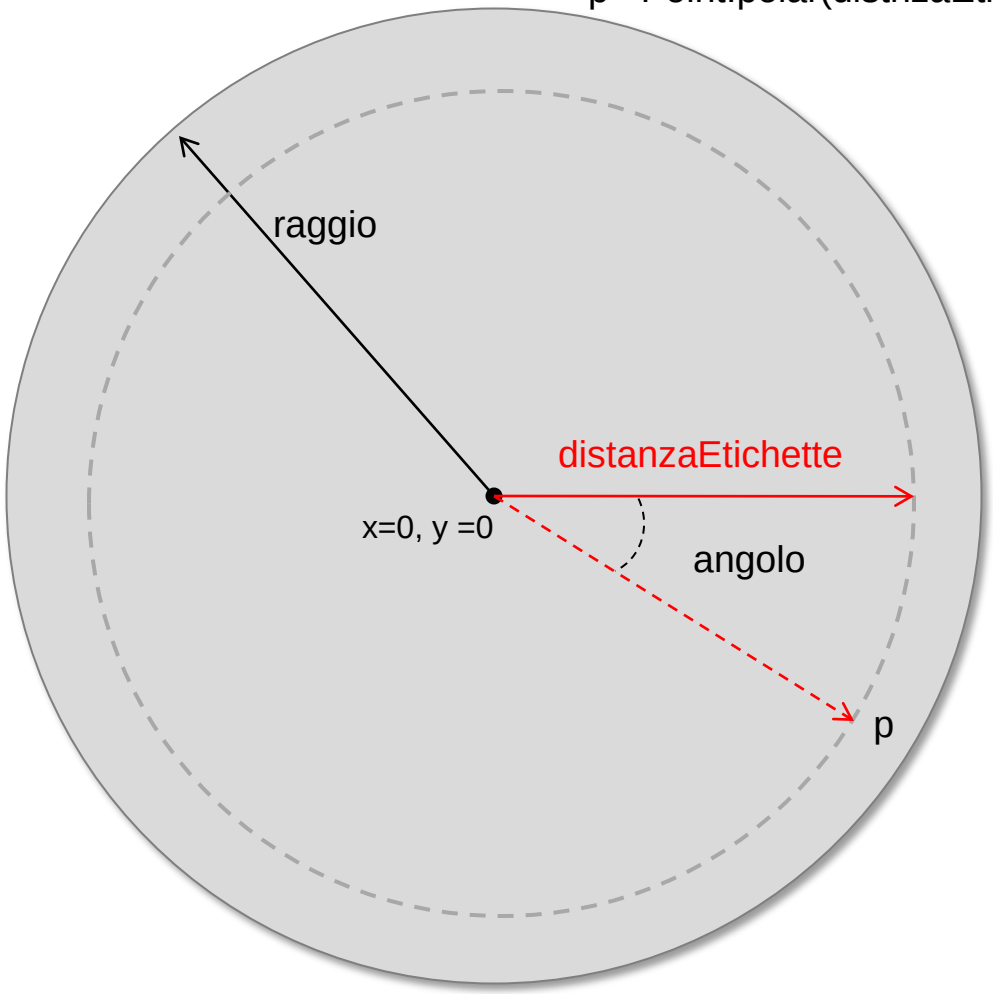
- La definizione di queste misure come proprietà e non come valori rende il mio lavoro più flessibile

```
private var lancettaOre:Shape = new Shape();
private var lancettaMinuti:Shape = new Shape();
private var lancettaSecondi:Shape = new Shape();
private var centrox:Number;
private var centroy:Number;
private var raggio:Number;

private var distanzaEtichette:uint;
```



p =Point.polar(distnzaEtichette, angolo)



quadrante.graphics.drawCircle(0,0,raggio)

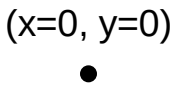
DISEGNARE LA LANCETTA

- Questa funzione disegna una lancetta sulla shape già creata:

```
private function disegnaLancetta(lancetta:Shape,
       colore:uint, lunghezza:uint, spessore:uint){
  lancetta.graphics.moveTo(0,0);
  lancetta.graphics.lineStyle(spessore, colore);
  lancetta.graphics.lineTo(0,lunghezza);
  addChild(lancetta);
  lancetta.x = centrox;
  lancetta.y = centroy;
}
```

DISEGNARE LA LANCETTA

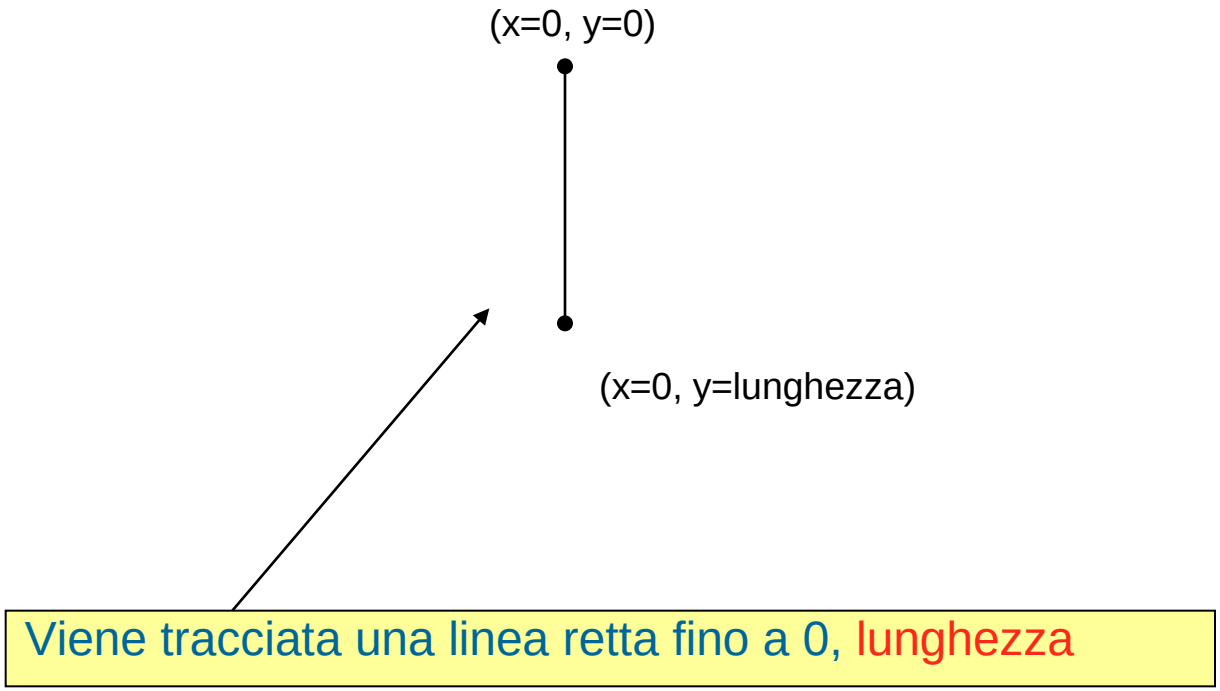
```
lacetta.graphics.moveTo(0,0);
```



Il pennino viene spostate alle coordinate x=0, y=0 relative all'oggetto Shape creato

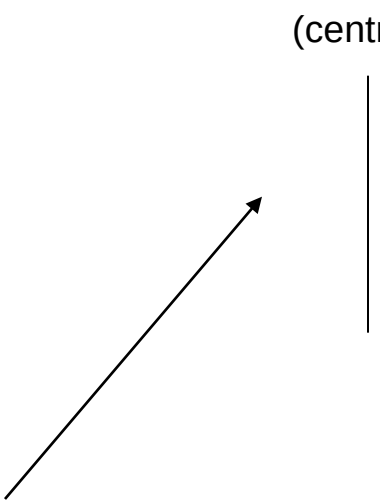
DISEGNARE LA LANCETTA

```
lancetta.graphics.lineStyle(spessore, colore);
lancetta.graphics.lineTo(0,lunghezza);
```



DISEGNARE LA LANCETTA

```
addChild(lancetta);
lancetta.x = centrox;
lancetta.y = centroy;
```



La lancetta viene aggiunta allo schermo e posizionata al centro dell'orologio pronta per essere ruotata dall'evento timer