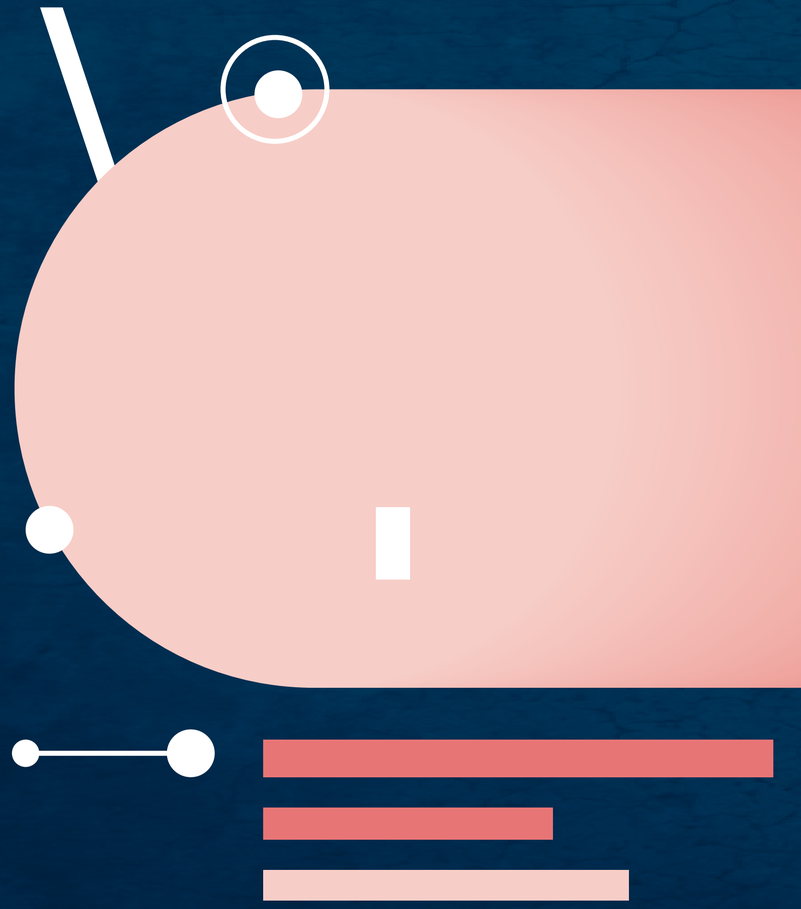
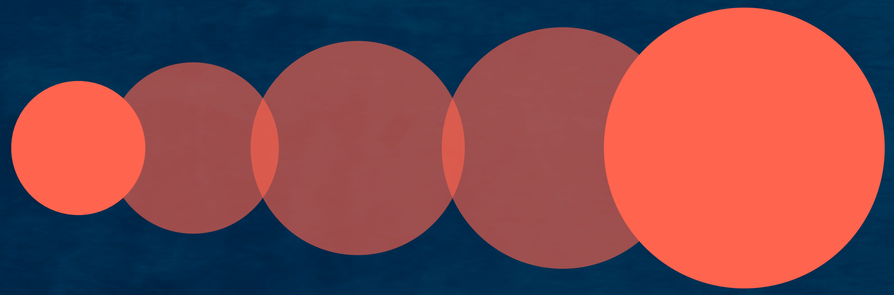
# 速度曲線與動畫控制

Canvas與特效動畫



# 連續變化的控制

- 製作動畫時常需要使用數值過渡
- 數值隨著時間變化到目標
- e.g. 爆炸波、角色移動、分數增加



r=10
left=0

r=20
left=100

# 連續變化的控制

爆炸波

角色移動

0 → 100

分數增加

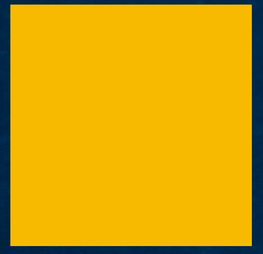
# 數值變化可控制的部分

變化的速度與形式

過程秒數

# Easing函數-動態

線性
-

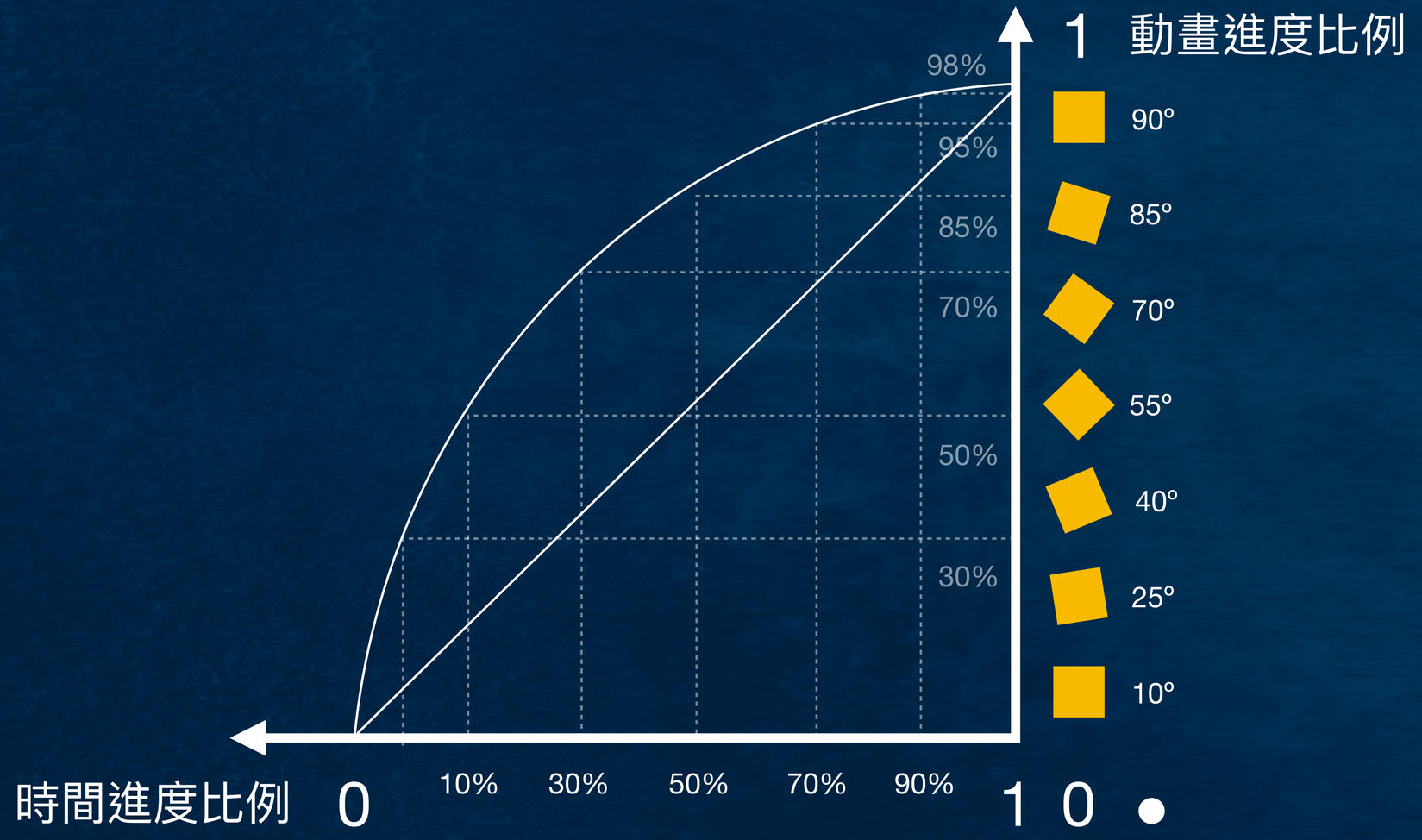


緩進
Ease in

緩出
Ease out

緩進出
Ease in out

# Easing函數-進度轉換



# Easing函數-控制速度曲線

時間 → 0-1 →  → 0-1 → 目標值

進度比例

進度比例

轉換成進度輸出

- 經過轉換輸入進度 / 將輸出轉換為對應數值
- 例如： 要y=50-150 ，y = 50 + (150-50)*進度比例

# 速度函數的函式庫

- 幾種常見的：

  次方(Power)
  彈性(Elastic)
  反彈(Bounce)

- 類型：

  easeOut 慢出
  easeIn 慢進
  ease 慢進出

- // Elastic.easeOut

# 動畫操作的函式庫- TweenMax

- 很容易製作動畫
- 可以拿來操作數值
- 有很多內建的速度函數

# 使用TweenMax操作CSS

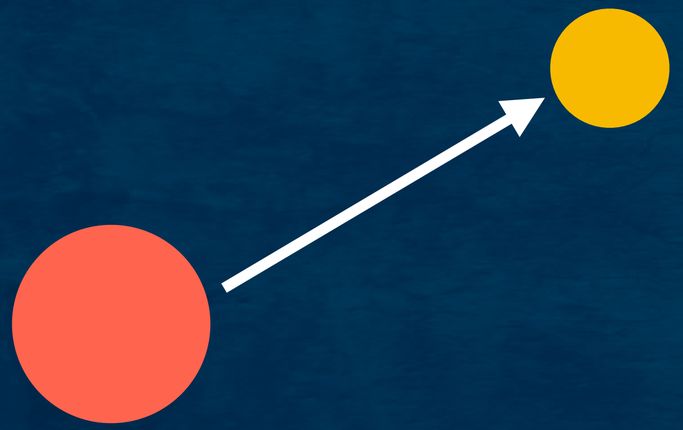
TweenMax.to ( obj , duration , otherProps )

Obj 要操作的物件

Duration 變化時間（秒）

otherProps 變化設定 e.g. 屬性 / 速度曲線 / delay…

e.g. TweenMax.to( “.ball”, 1, { left: 100, ease: Elastic.easeOut } )



# TweenMax Tips

```
{
    repeat:  3,                 // 重複三次
    yoyo: true,                 // 重複時偶數反轉
    yoyoEase: Power2.easeIn   //yoyo時速度函數
    delay: 1                    // 延遲時間
    ease: Elastic.easeIn      //速度函數
}
```

變化使用css時使用 camelSize  // backgroundColor

Transform 轉換: x , y , z , scale , rotation

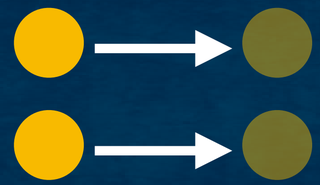
# TweenMax 函數

1. 變化到指定狀態

.to( obj , duration , otherProps )

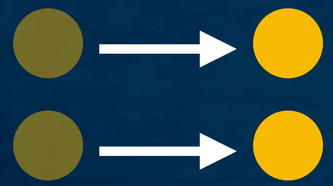
2. 從指定狀態變化到當下

.from( obj , duration , otherProps )



3. 多物件 / 依序變化到指定狀態

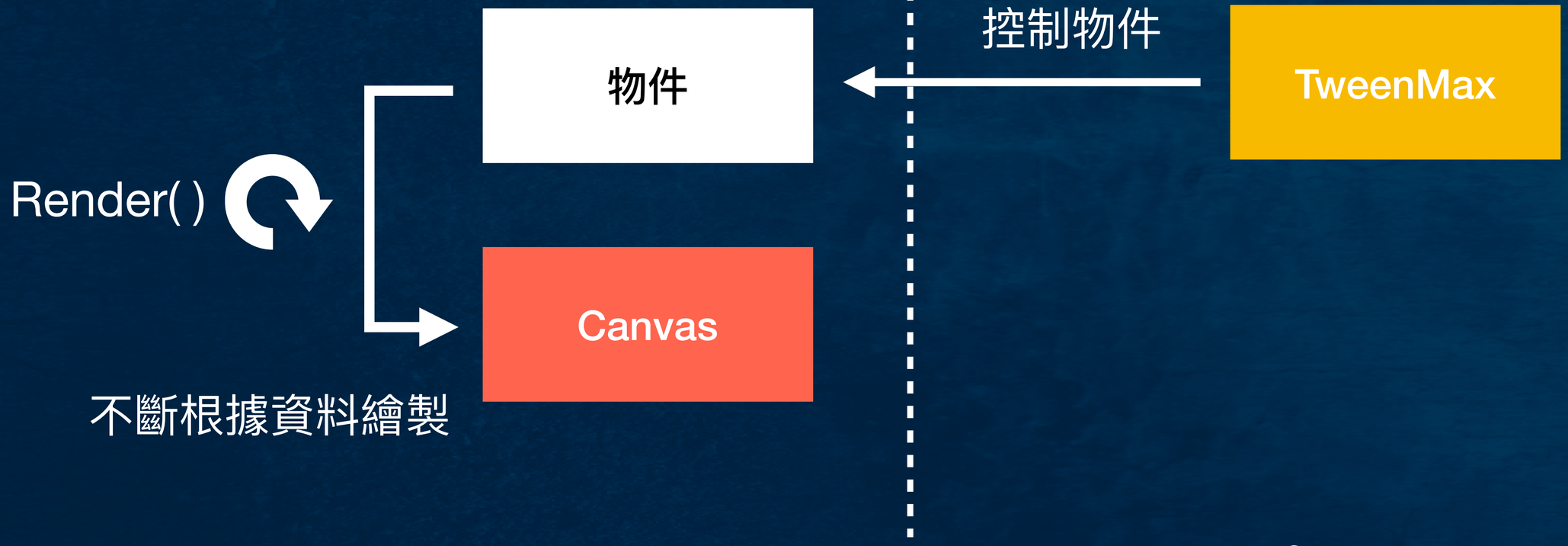
.staggerTo([obj], duration , otherProps , spaceTime)



4. 多物件 // 依序從指定狀態變化

.staggerFrom([obj], duration , otherProps , spaceTime )

# 使用TweenMax操作數值與物件



# 使用TweenMax操作數值與物件

- 將html元件替換成物件
- 指定屬性值的目標操作
- onUpdate更新值觸發

```
var obj = {
  value : 0
}
TweenMax.to(obj,1,{
  value: 100,
  onUpdate: function(){
    console.log(obj.value) //1.2.3...
  }
})
```

# 實作範例練習

# 繪製一個小城市與動態