**abstract class 和 interface 的區別**  
  
**abstract class的特徵:**  
1. abstract class和abstract method都必須用abstract關鍵字來修飾  
2. abstract class不能用new關鍵字去產生物件  
3. abstract method只需要宣告, 不需要實作  
4. 繼承abstract class的子類別必須實作父類別的abstract method, 否則這子類別還是個abstract class  
  
**interface的特徵:**  
1. interface沒有建構方法 (即method中沒有參數, 且沒有任何變數的設定)  
     Ex: interface Bus{  
                void sound();  
           }  
2. interface中所有資料成員都必須初始化, 且均為常數  
3. 宣告變數必須是final, static和public  
4. interface中的method必須為abstact或public

**abstract class 和 interface 的使用時機**

何謂abstract class?

供一種多個class一起合作工作的方式, 將多個class中相同的元素pull up method到public class中, 再以繼承的方式來使用它, 目的是為了實現"多型"精神

(註) 一個子類別只能繼承一個父類別, 但一個父類別可以被許多子類別所共同繼承

何謂interface?

即spec., 完全不需要定義實作, 只需要函式原型  
若要實作interface, 就必須follow它的spec.

[public] [abstract] interface 介面名稱{

權限設定  傳回型態  method(parameters);

權限設定  傳回型態  method(parameters);

}

[public] [abstract]是預設, 所以可省略, 因為interface本身就是抽象的  
(註)一個介面可以同時繼承多個介面, 即同時繼承了多個介面的abstract method和常數  
 => interface A **extends** 介面1, 介面2, 介面3, ...

一個class可以同時實作多個interface  
 => class B **implement**s 介面1, 介面2, 介面3, ...

---------------------------------------------------------------------------------------------------

**abstract 和 interface 的使用時機個人結論**

**abstract:**

1. 特徵相同，可以設定成共同屬性，
2. 若子類別需要做相同的事，則可以在抽象類別設定方法
3. 在子類別中，各自名稱相同但內容不同的方法
4. 避免抽象類別被實體化，因為抽象類別只是共同的特徵

**Interface:**

1. 介面像是一個規範，所以實作的類別都要遵守這個規範，而不論繼承的類別實作方式是什麼。
2. 需要遵守多個規範時，可以以介面方式來實作

參考資料:

http://jimmu-jimmu.blogspot.com/2012/10/abstract-classinterface.html