|  |  |  |
| --- | --- | --- |
|  | java | kotlin |
| 宣告  變數 | 有宣告變數或常數(final)  還要宣告char 和int 等等屬性 | 只需要宣告var(變數)或val(常數) (至於是int 或是char等等….kotlin都會自行偵測) |
| case | switch(I){  case (I.1):  …. = A;  break;  case (I.2):  …. = B;  break;  } | when(I){  I.1 -> …. = A  I.2 -> …. = B  } |
|  | 有new | 沒有(不用)new |
| Override | @Override  函式() | override fun () |
| Override(覆寫)子類別繼承父類別時，改寫父類別原有的方法內容。 | | |
| Method(方法) | 修飾詞 資料返回類型 方法名稱(參數資料類型 參數名稱){  //程式碼執行區塊  } | 修飾詞 fun 方法名稱(參數資料類型 參數名稱){  //程式碼執行區塊  } |
|  |  |  |

心得:

現在java雖然是現在最受歡迎的程式語言之一，但是我覺得他的物件導向好複雜，要記得很多物件要如何使用和參數要如何配置等等……，kotlin也全都是物件，在這麼方便的語言下需要什麼樣的功能就呼叫特定的物件，與我之前寫程式的環境不同，要甚麼功能就得先自己顯出一個class自己寫function，可能是寫程式突然變得那麼方便所以才讓我有點不適應吧。

雖然C++也是物件導向程式語言，但他提供的物件功能不向Java那麼豐富所以才不像Java那麼多人喜歡，我也知道為了設計程式的流暢與降低程式出錯物件導向是如此的方便。

有個讓很多java工程師頭痛的bug，就是NPE(null\_Pointer\_Exception)，

但是在kotlin中已經完美的解決了該問題，只需一個問號：”？”，而NPE

意旨:基本的JAVA數據類型，變量的值要麼已經是默認值，如果沒有對其正常賦值，程序便不能通過編譯。