



# 輕鬆完成一個 Undoable 應用程式

侯捷

自由寫譯、資訊培育、大學任教









( 金色陽光, 活力Java  
Java 2006 專業技術大會

- Undo/Redo 功能觀察
- Java **Undoable** classes 檔案組織
- **Undoable** classes 架構與運行
- 寫出你自己的 **Undoable** 應用程式



# 前言

- **Command** pattern 適合實現 undoable 功能
- Java Swing 有一組 **Undoable** APIs 與之對應
- 這是一組龐大的 classes，組織繁複，便利性高
- 本講題探討 Java Swing **Undoable** APIs 的  
組織、架構、運行、應用



# Undo/Redo 觀察 (1 for Microsoft Word)

- 輸入"abc" 後按  則 "abc" 都消失
- 輸入"a b c" 後按  則 "a b c" 都消失
- 輸入"a  b  c " 後按  仍是一次性全部消失。
- 若在輸入"abc" 過程中每輸入一個字元就將游標移走再移回，則  以字元為單位。
- 即使輸入是正確的英文語句，行為模式不變。
- 造成文件內容（文字、格式）改動的任何動作都被視為 significant（有重大意義），可undo/redo。其他動作（如圈選、移動游標）被視為insignificant，不可undo/redo。

## Undo/Redo 觀察 (2 for Microsoft Word)

- 以注音法輸入中文（永遠一次一字），則  以字為單位。
- 以詞庫法輸入中文，則  以詞為單位，因每次輸入類似 copy/paste。
- 注音法的智慧型加選部分視為一個undo/redo單位，因它是 copy/paste 產物。

# Undo/Redo 觀察 (3 for Microsoft Word)

- UI 會呈現 undoable/redoable 操作的名稱（例如 "鍵入"、"清除"、"雙刪除線"、貼上格式"、"字型顏色"...）及 prefix（例如 "復原"、"取消"、"重複"）。
- 存在多個 undoable/redoable 操作時，  會條列每個操作供用戶選擇。
- 可選擇先前某個（不必是前一個）undoable/redoable 操作，一步到位完成所有 undo/redo。



復原  
復原鍵入  
復原清除  
復原雙刪除線  
復原貼上格式



取消復原  
取消復原鍵入  
取消復原清除  
取消復原雙刪除線  
取消復原貼上格式

## Undo/Redo 觀察 (4 for Microsoft PowerPoint)

- "物件群組"、"放映方式"、"投影片切換方式" 都可以 undo/redo。因為這些操作都改動了文件內容。
- 欲知某個操作能否 undo/redo，只要在操作之後看看 "undo 清單" 中有無對應項目（應為最新一個）即知。

# Undo/Redo 觀察 (5 for Microsoft 小畫家)

- "小畫家" 的 undo/redo (復原/重複) 只記錄最新三個操作。



# Command 範式

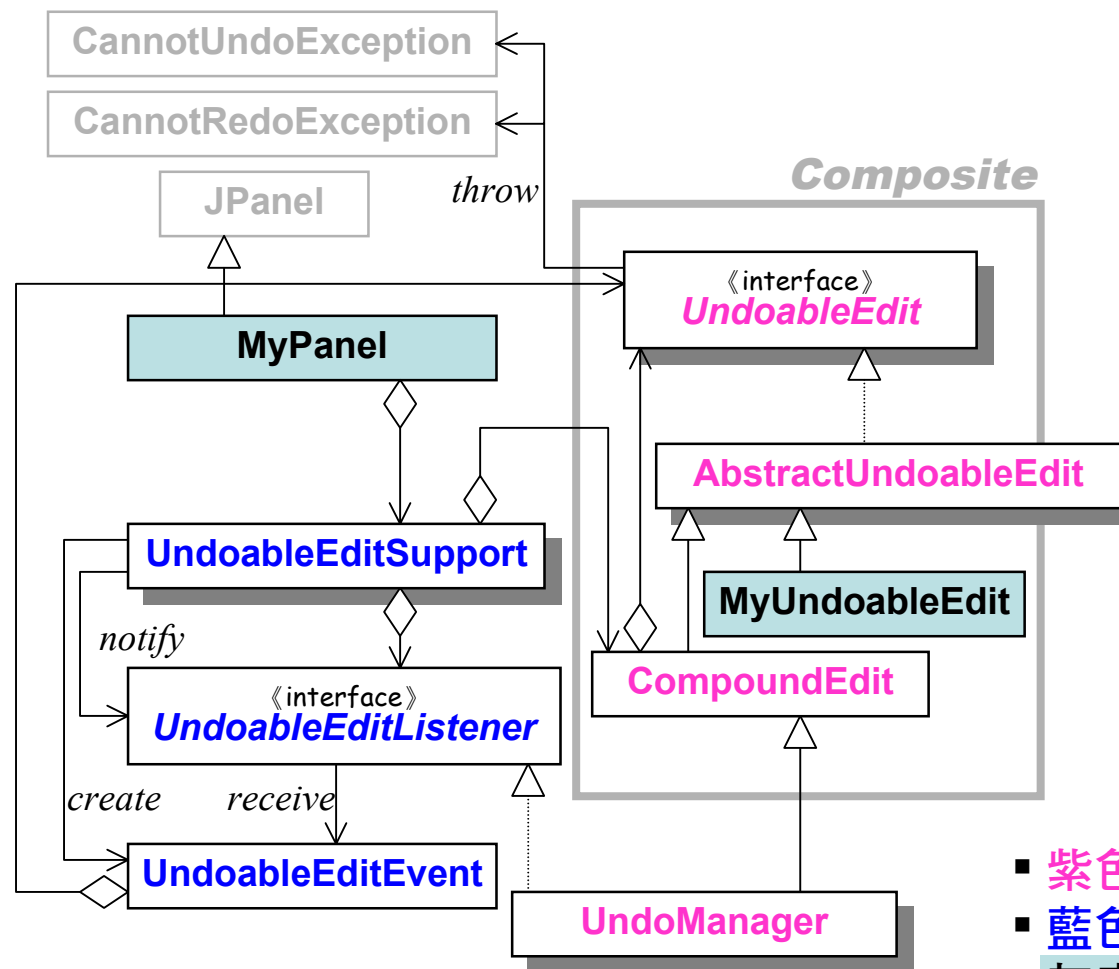
- Encapsulate a request as an object, thereby letting you parameterize clients with different requests, queue or log requests, **and support undoable operations.**
- 將 request 封裝為 object，讓你得以不同的 requests 對 client 參數化，或將 requests 放進 queue 中或作為日誌（log）保存下來，**並支援 undoable 操作。**

# Undoable APIs 檔案組織

src\javax\swing\undo	src\javax\swing\event	src\javax\swing	src\javax\swing\text
<b>AbstractUndoableEdit</b>	UndoableEditEvent	JTextArea	<b>AbstractDocument</b>
CannotRedoException	UndoableEditListener		<b>Document</b>
CannotUndoException			<b>GapContent</b>
<b>CompoundEdit</b>			<b>JTextComponent</b>
* StateEdit			<b>PlainDocument</b>
* StateEditable			
<b>UndoableEdit</b>			
UndoableEditSupport			
<b>UndoManager</b>			

- 紫色表示 fundamental classes
- 藍色表示讓編程更方便的classes。
- 紅色表示 JTextArea背後用以支援 undo/redo 的 classes

# Undoable APIs 架構 (1)

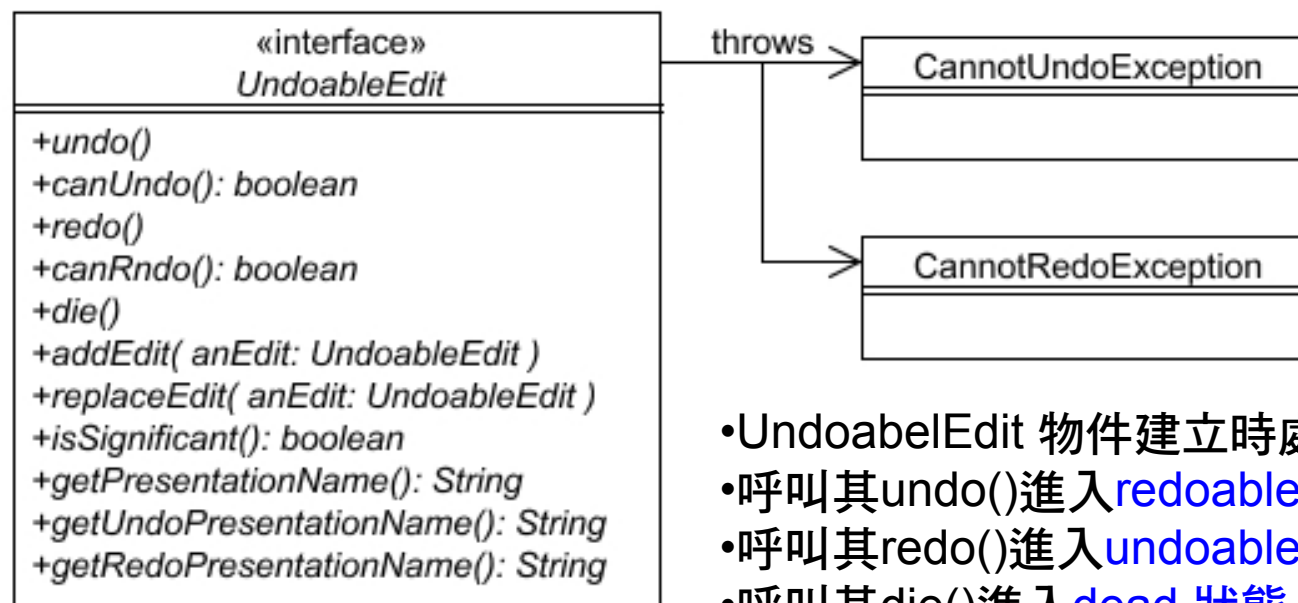


應用程式中直接用到  
`UndoableEdit`,  
`UndoManager`,  
`UndoableEditSupport`

繪圖程式例中 `main()` 產生  
`MyPanel` 和 `UndoManager`  
物件；運行中不斷產生  
`MyUndoableEdit` 物件

- 紫色：fundamental classes
- 藍色：讓編程更方便的輔助性 classes
- 灰底：應用程式的 classes

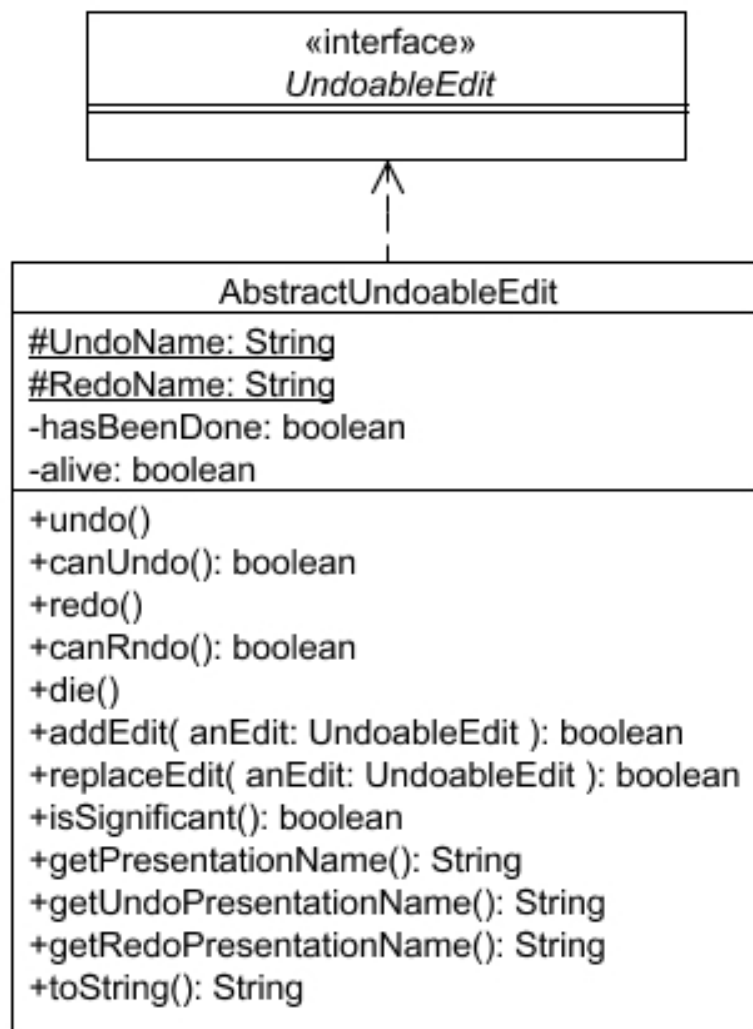
# Undoable APIs 架構 (2. UndoableEdit)



- UndoableEdit 物件建立時處於undoable狀態
- 呼叫其undo()進入redoable狀態
- 呼叫其redo()進入undoable狀態
- 呼叫其die()進入dead 狀態
- addEdit()和replaceEdit()提供物件合併相關操作
- isSignification()傳回物件「是否有意義」

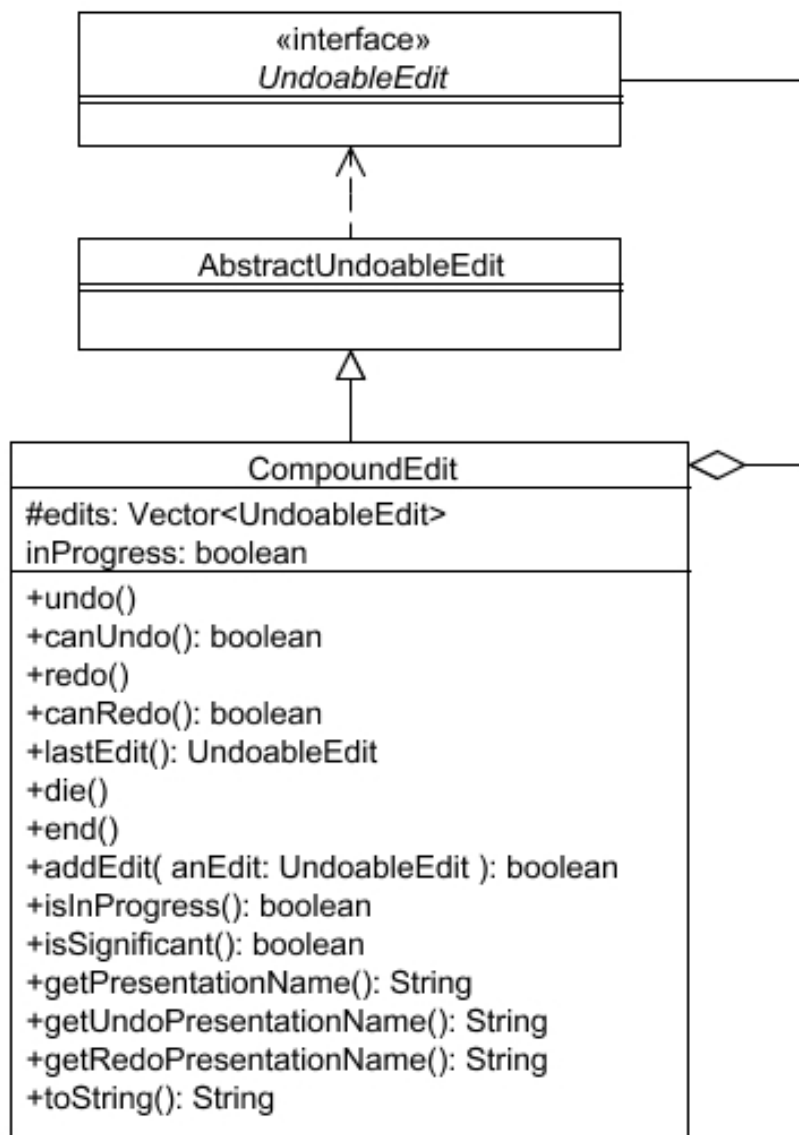
- getPresentationName()傳回物件代表字串，以文字編輯而言可能是 "剪下"、"貼上"、"鍵入"、"刪除" 等等。
- getUndoxxx()和getRedoxxx()將上述所得名稱加上 prefix ("復原" 或 "取消")，所得文字可顯示於 UI 提示用戶。

# Undoable APIs 架構 (3. AbstractUndoableEdit)



- 以兩個 flags (`alive`、`hasBeenDone`) 實現 undoable/redoable/dead 狀態模型
- 兩者初值皆為 `true`。
- 呼叫 `undo()` 會把 `hasBeenDone` 設為 `false`
- 呼叫 `redo()` 會把 `hasBennDone` 設回 `true`
- 呼叫 `die()` 會把 `alive` 設為 `false`
- 編寫吾人自己的 Undoable 程式時，只要繼承 `AbstractUndoableEdit` 並覆寫 `undo()`、`redo()` 和 `die()`，並在其中首先呼叫 `supre.undo()`、`super.redo()` 和 `super.die()`，就無需操心「狀態切換」。

# Undoable APIs 架構 (4. CompoundEdit)



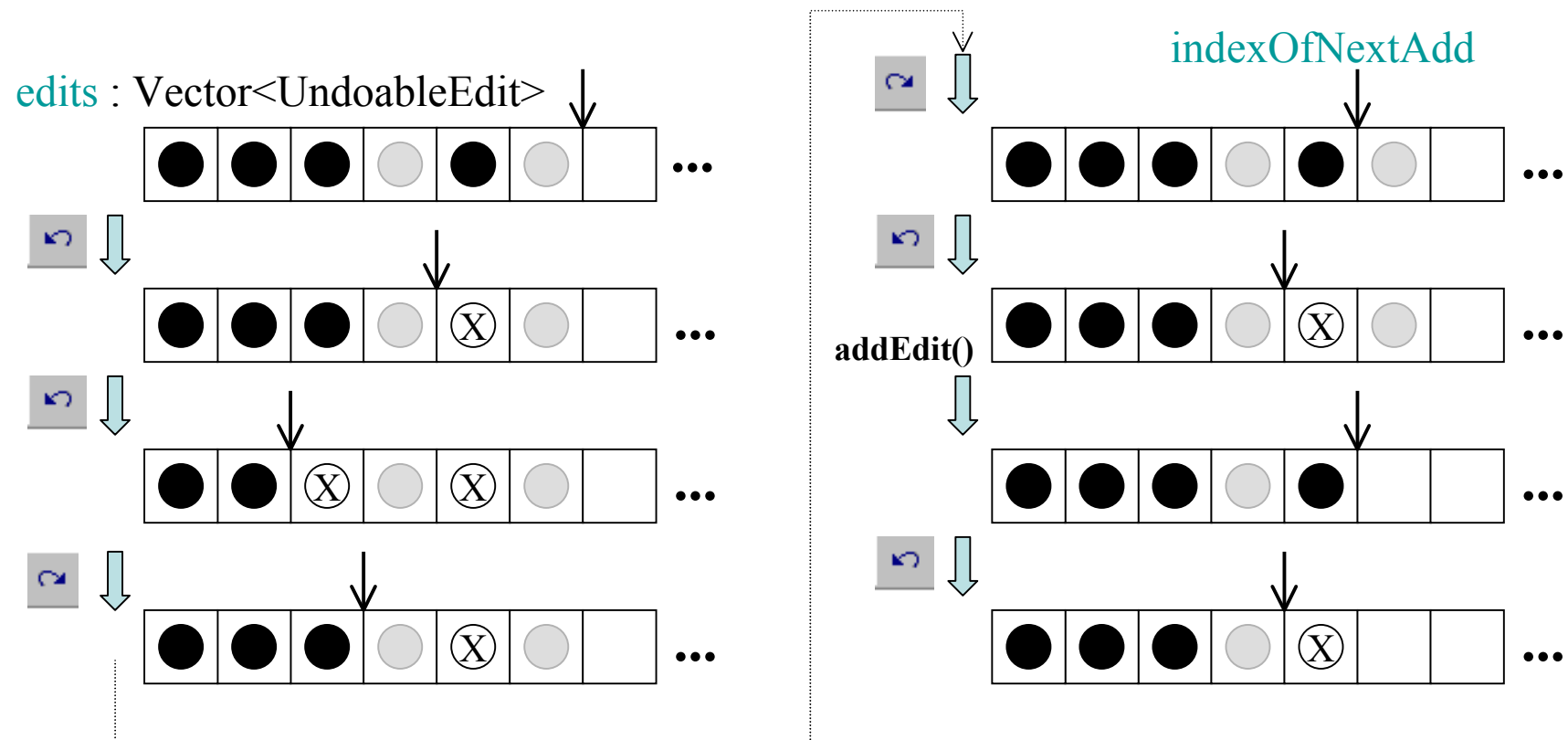
- 可將多個 Undoable 操作合而為一。例如「連續輸入字元」合併為「輸入一個字串」，undo時一次還原整個字串。
- 內有一個 UndoableEdit list (**edits**) 和一個初值為 true 的 flag **inProgress**：true 表示正在建立一個 CompoundEdit，此時不能 undo/redo
- 呼叫 addEdit() 會將傳入的 edit 加入 list
- 呼叫 end() 表示合併完成，於是 **inProgress** 被設為 false，此後整個 CompoundEdit 的行為如同單一 UndoableEdit。

# Undoable APIs 架構 (5. UndoManager)

UndoManager
indexOfNextAdd: int limit: int
+getLimit(): int +setLimit( l: int ) +discardAllEdits() #trimForLimit() #trimEdits( from: int, to: int ) #editToBeUndone(): UndoableEdit #editToBeRedone(): UndoableEdit +undoTo( edit: UndoableEdit ) +redoTo( edit: UndoableEdit ) +undoOrRedo() +canUndoOrRedo(): boolean +undo() +canUndo(): boolean +redo() +canRedo(): boolean +lastEdit(): UndoableEdit +end() +addEdit( anEdit: UndoableEdit ): boolean +getPresentationName(): String +getUndoPresentationName(): String +getRedoPresentationName(): String +undoableEditHappened() +toString(): String

- 顧名思義，是個管理器，負責管理誰（哪一個 UndoableEdit）將被undo/redo。
- 對 undoable list 長度有所限制，記錄於int limit，預設100。undoable list 長度達上限時會採取必要措施。
- 實作了UndoableEventListener介面，因此可成爲 UndoableEditEvent 的接收器，或說成爲 UndoableEditEvent 製造者的監聽器。
- undo() 被呼叫後首先以 editToBeUndone() 從 list 中找到最後的一個 undoable significant UndoableEdit，然後從 indexOfNextAdd 所指位置起至該 UndoableEdit 止全部 undo (使用undoTo())。

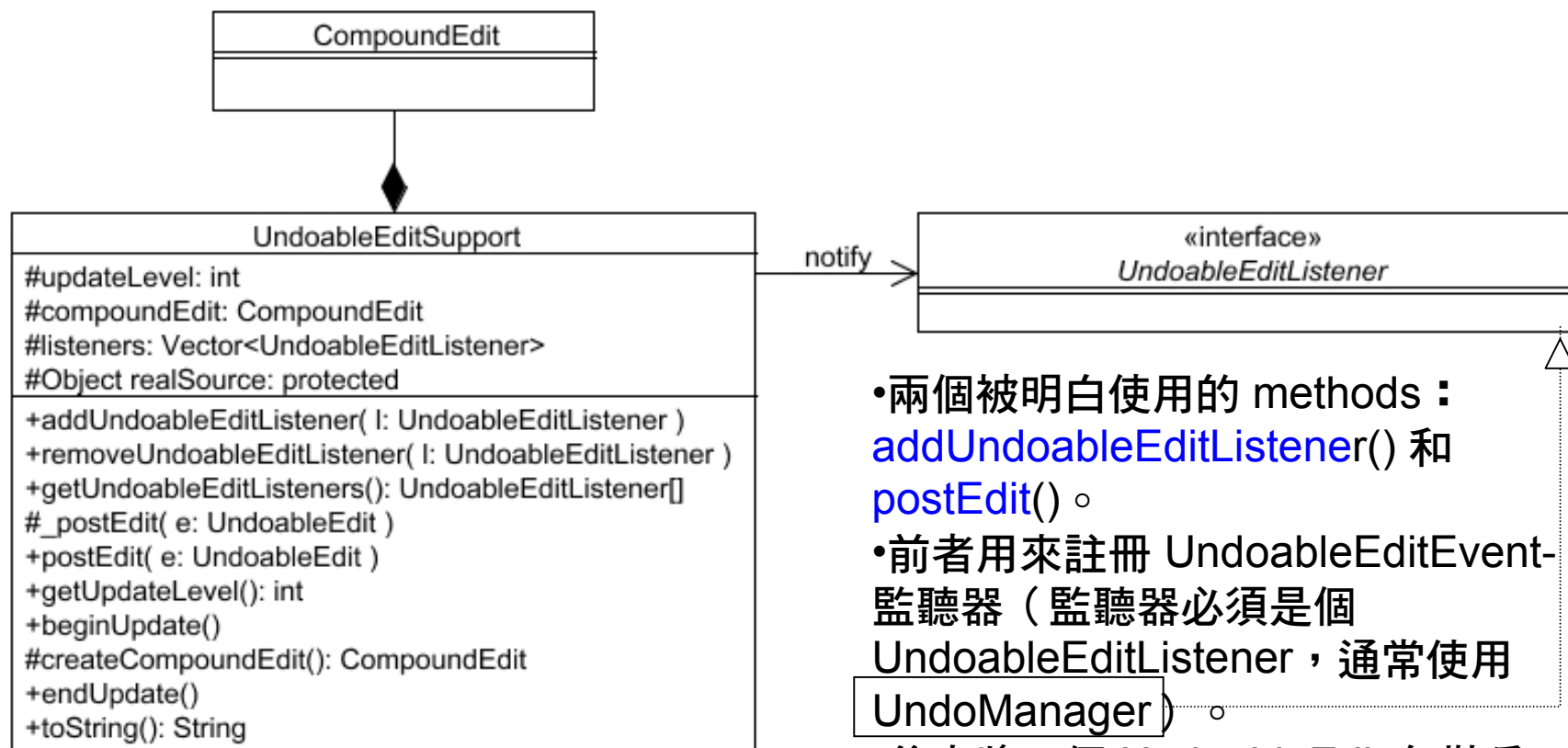
# Undoable APIs 架構 (5. UndoManager)



● : undoable significant edit  
⊗ : undone (redoable) significant edit  
○ : insignificant edit

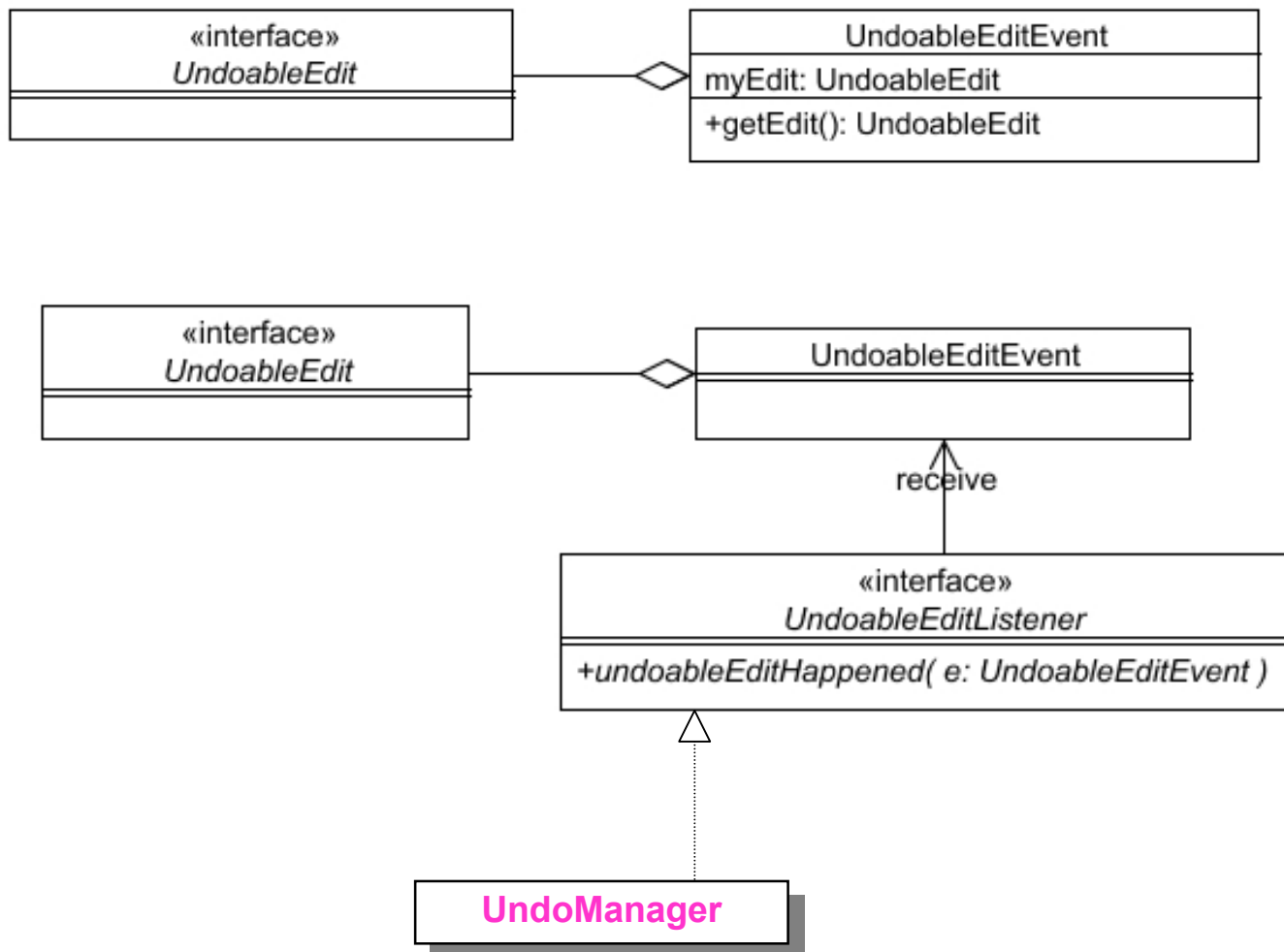


# Undoable APIs 架構 (6. UndoableEditSupport)



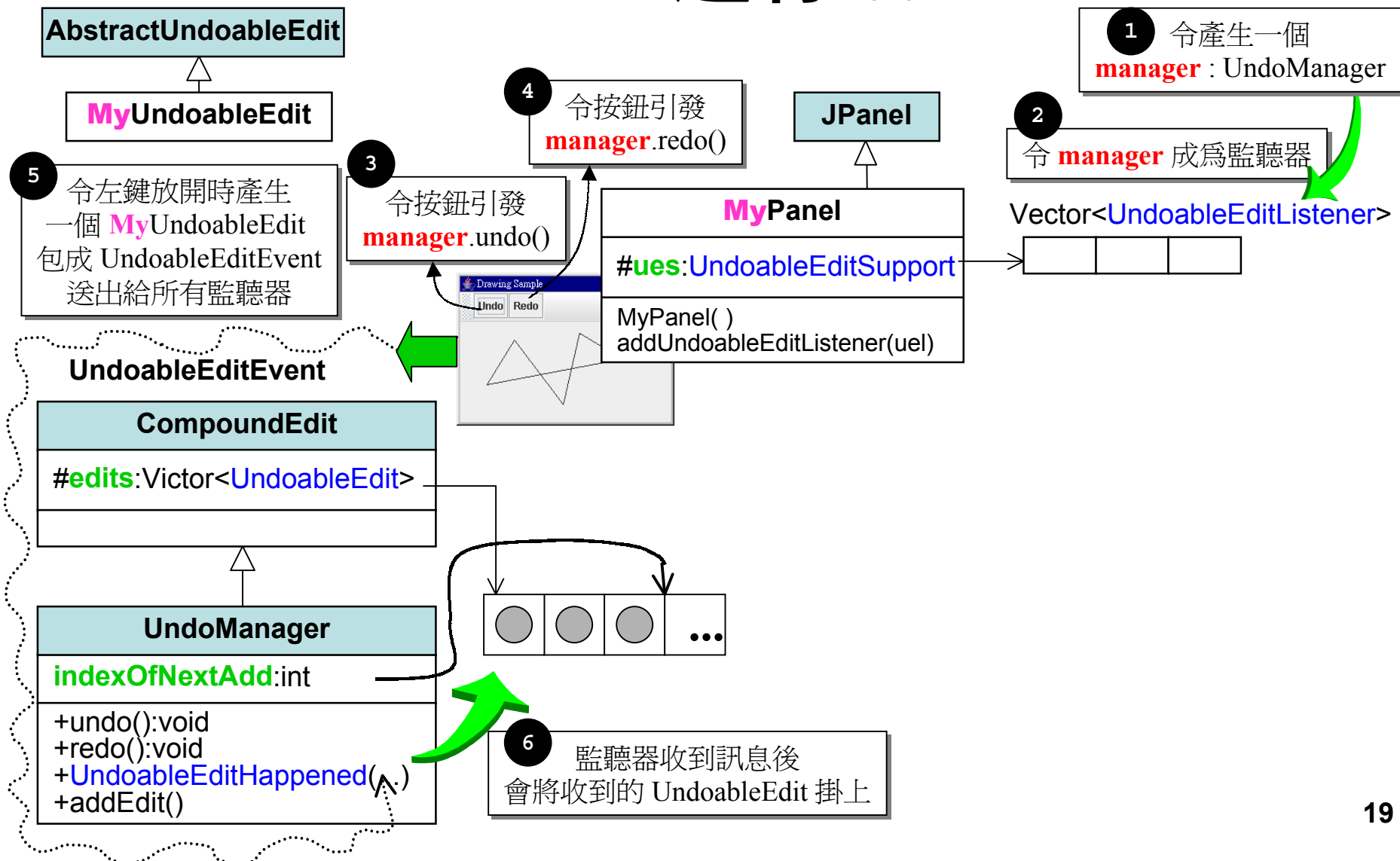
- 兩個被明白使用的 methods : **addUndoableEditListener()** 和 **postEdit()**。
- 前者用來註冊 UndoableEditEvent-監聽器 (監聽器必須是個 UndoableEditListener, 通常使用 **UndoManager**)。
- 後者將一個 UndoableEdit 包裝為 UndoableEditEvent 訊息並將之發送給所有監聽器。

# Undoable APIs 架構 (7. UndoableEventListener)

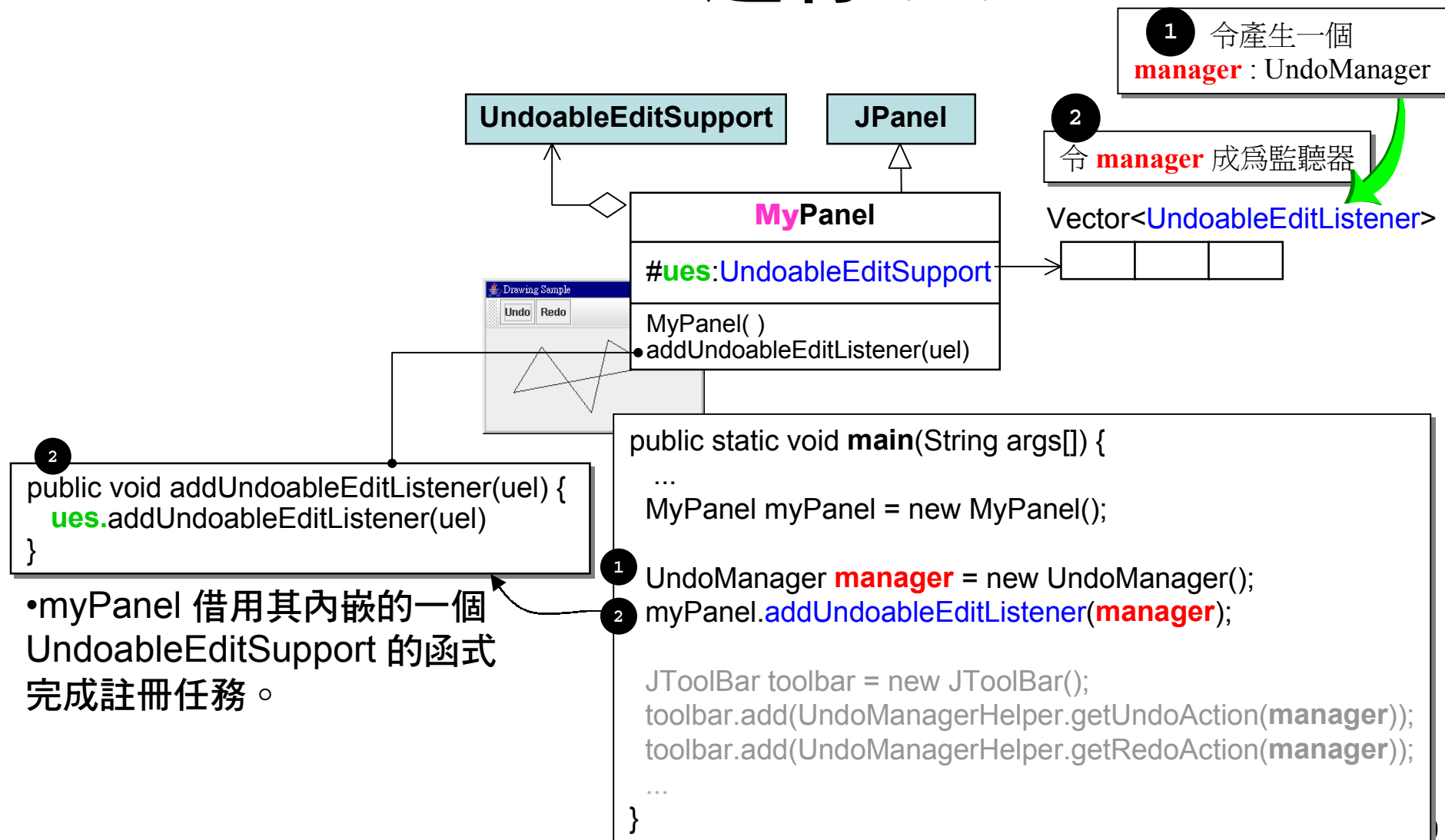


例見 <http://www.java2s.com/Code/Java/Swing-JFC/UndoableDrawingPanel2.htm>  
 《Definitive Guide to Swing for Java 2》2E, By John Zukowski. Publisher: APress

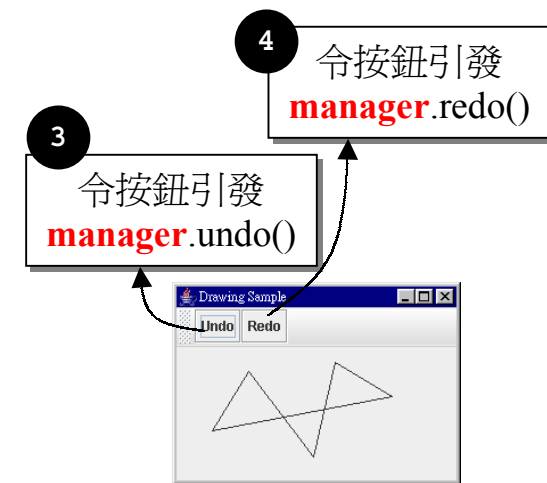
# Undoable APIs 運行 (0)



# Undoable APIs 運行 (1,2)



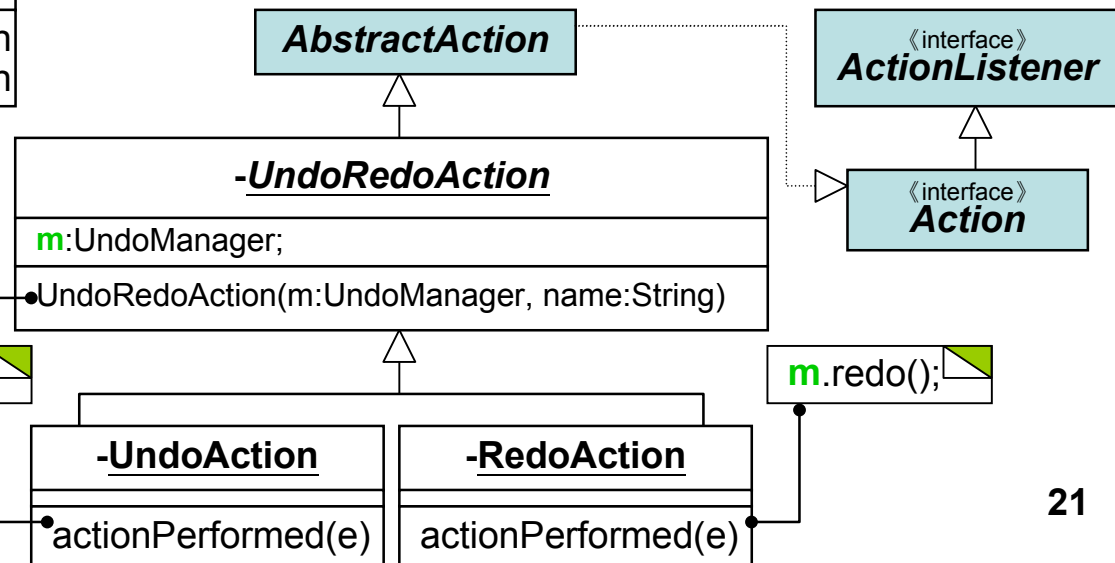
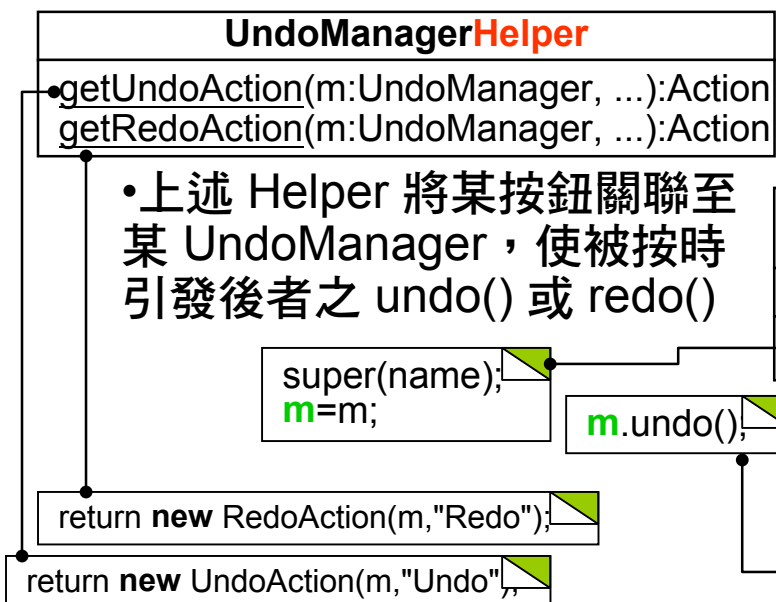
# Undoable APIs 運行 (3,4)



```
public static void main(String args[]) {
    ...
    MyPanel myPanel = new MyPanel();

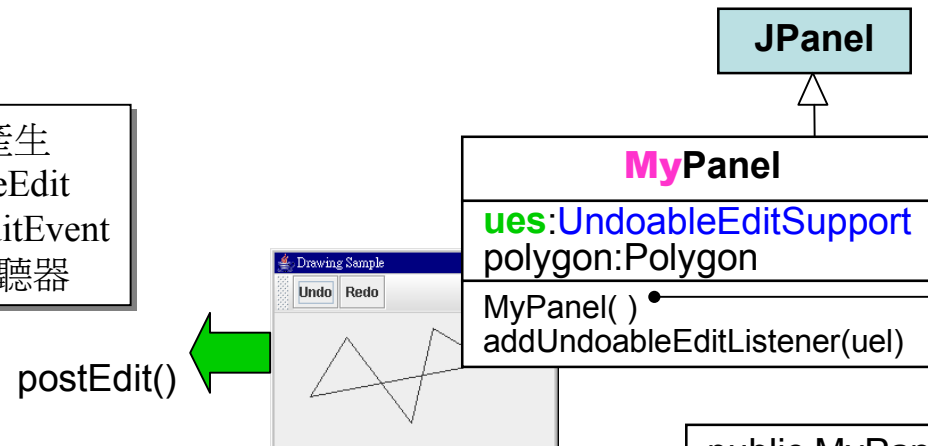
    UndoManager manager = new UndoManager();
    myPanel.addUndoableEditListener(manager);

    JToolBar toolbar = new JToolBar();
    3 toolbar.add(UndoManagerHelper.getUndoAction(manager));
    4 toolbar.add(UndoManagerHelper.getRedoAction(manager));
    ...
}
```



# Undoable APIs 運行 (5)

5 令左鍵放開時產生一個 **MyUndoableEdit** 包裝成 **UndoableEditEvent** 送出給每一個監聽器

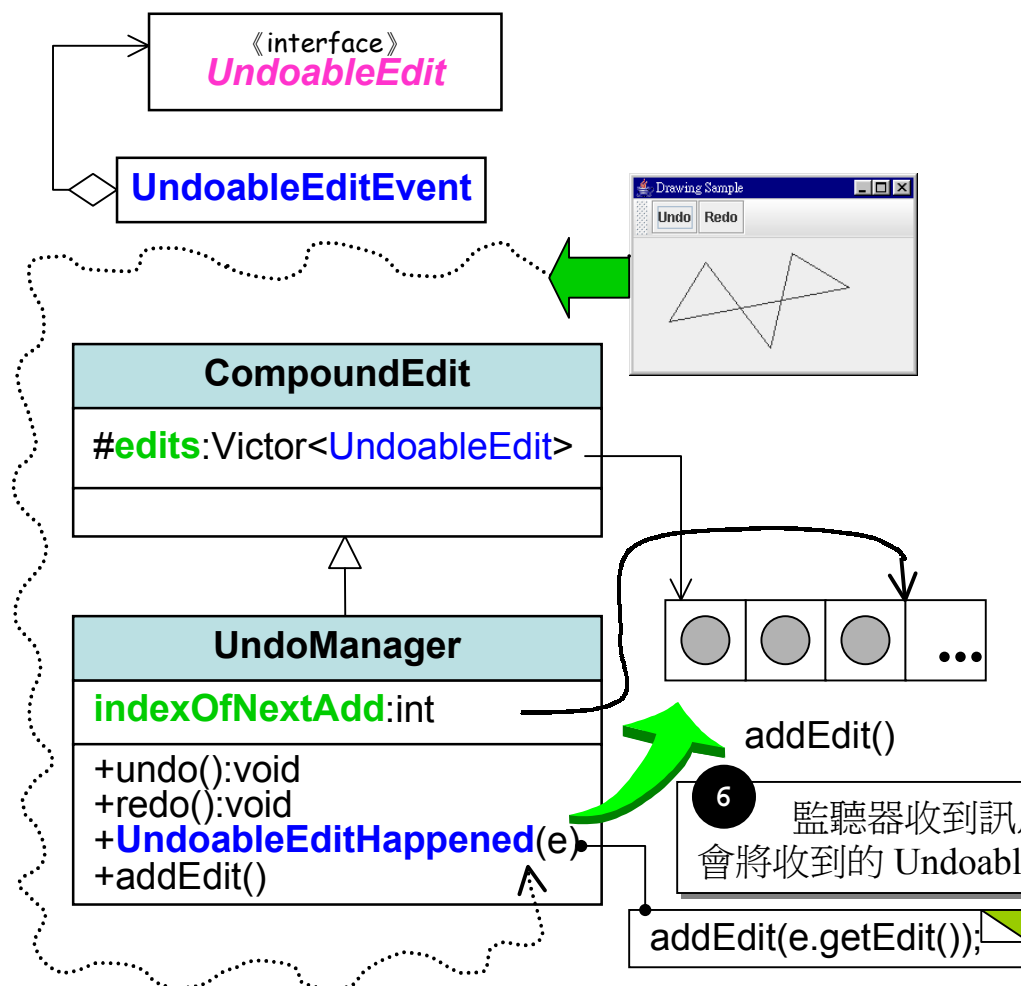


- MyPanel 必須監聽滑鼠左鍵鬆開動作，並於彼時借用內嵌之 UndoableEditSupport 的 postEdit() 完成傳送任務。
- 注意此例先創建 MyUndoableEdit（那將記錄現況）再為 polygon 添加新點。

```
public MyPanel() { //ctor
    MouseListener mL = new MouseAdapter() {
        public void mouseReleased(MouseEvent mE) {
            undoableEditSupport.postEdit(
                new MyUndoableEdit(MyPanel.this));
            polygon.addPoint(mouseEvent.getX(),
                            mouseEvent.getY());

            repaint();
        }
    };
    addMouseListener(mL);
}
```

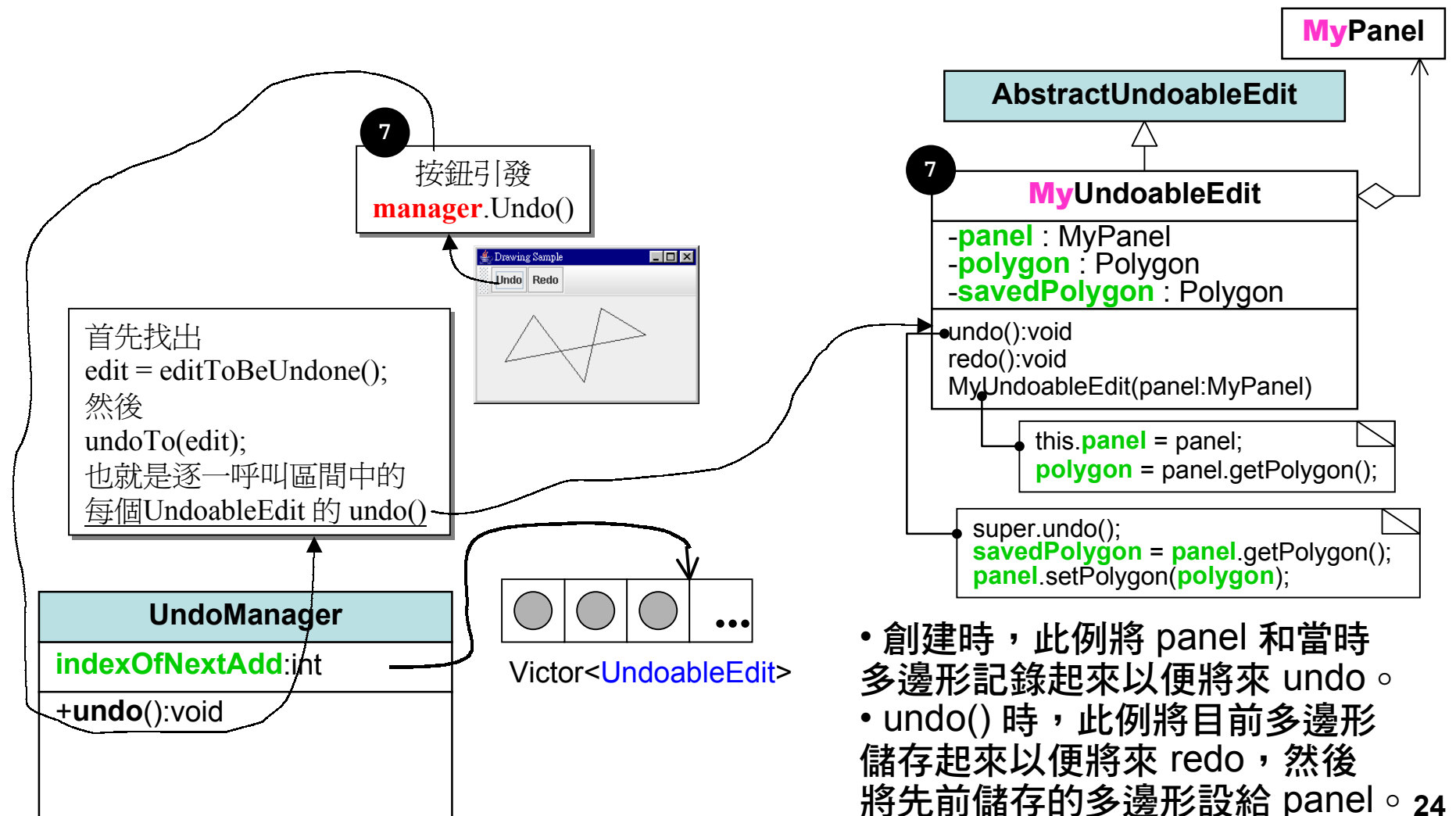
# Undoable APIs 運行 (6)



- 這個 UndoManager 先前曾經註冊成為監聽器

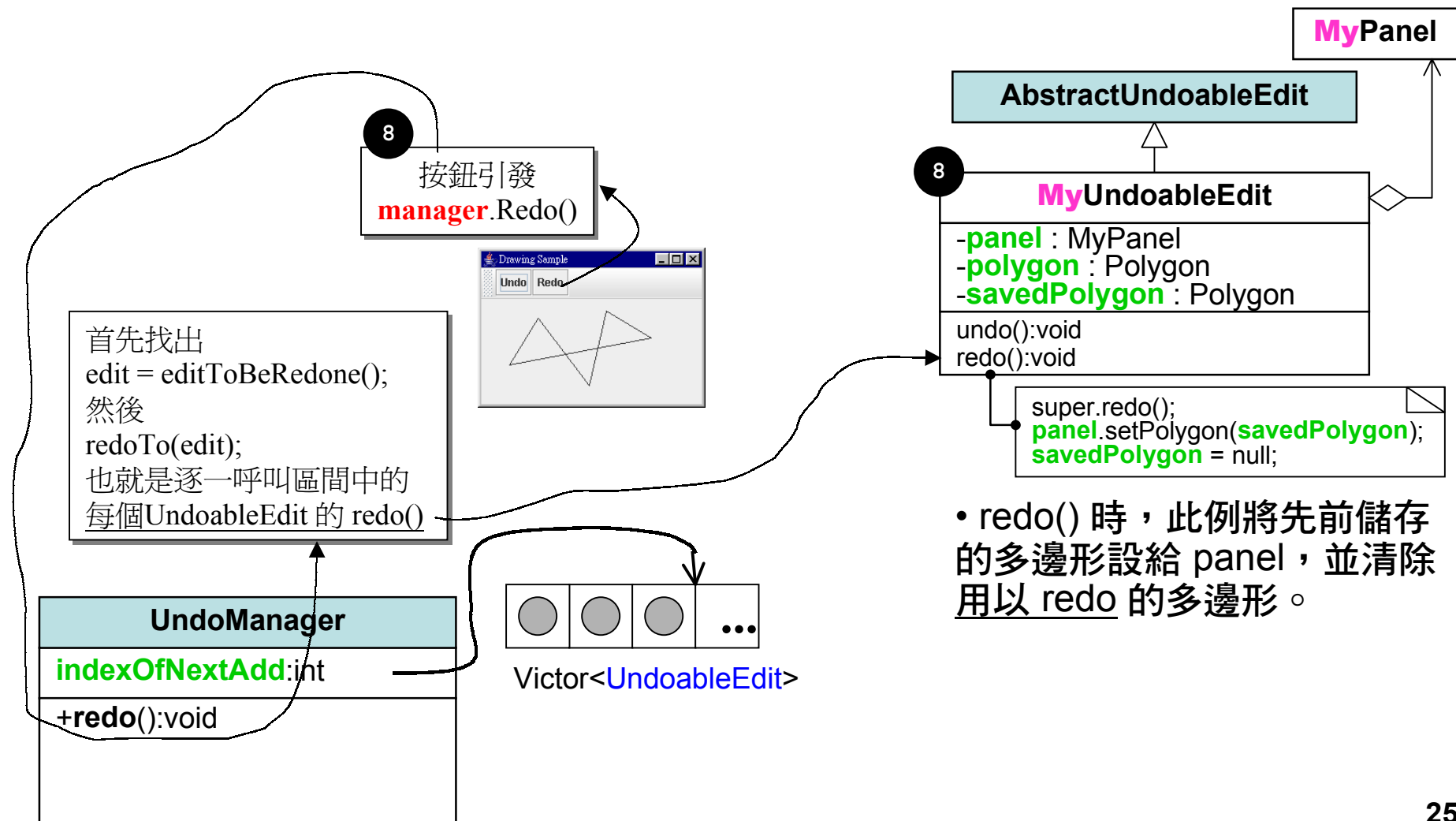
6 監聽器收到訊息後  
會將收到的 UndoableEdit 掛上

# Undoable APIs 運行 (7)



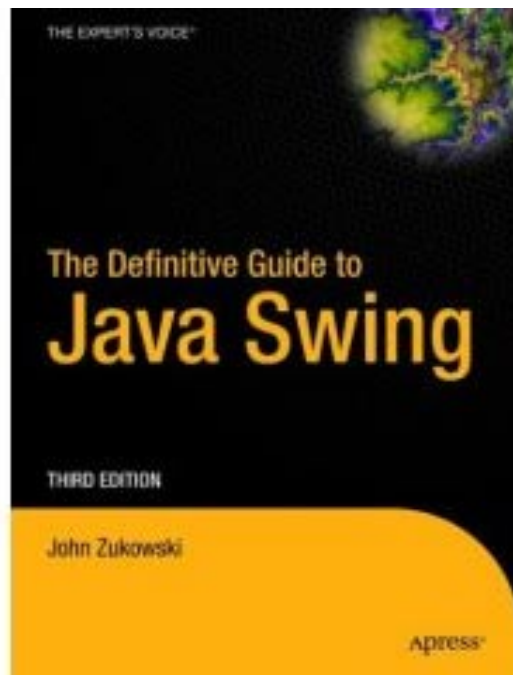


# Undoable APIs 運行 (8)

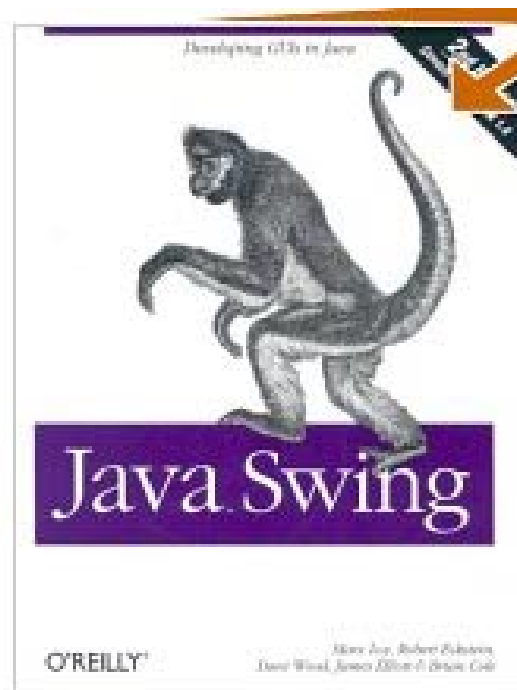


# 更多資訊

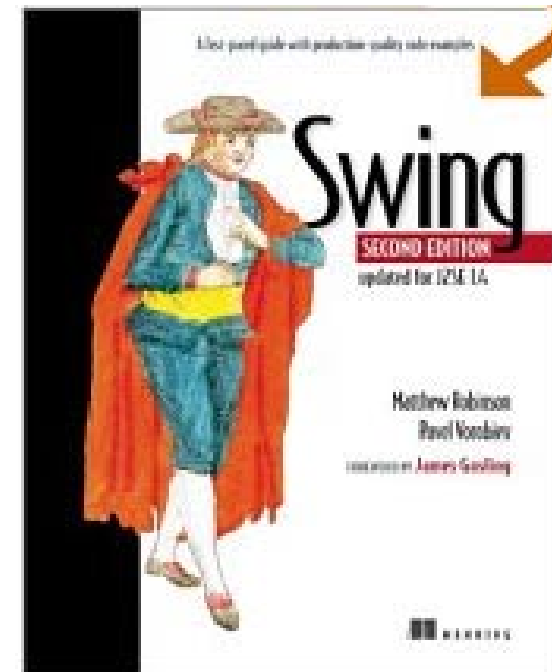
- "輕鬆完成一個 Java Undoable 程式", by 侯捷, [http://www.jjhou.com/...](http://www.jjhou.com/)



The Definitive Guide to Java Swing, 3e  
by John Zukowski  
Paperback: 928 pages  
Publisher: Apress (June 13, 2005)  
ISBN: 1590594479



Java Swing, 2e  
by James Elliott, Robert Eckstein (Editor),  
Marc Loy, David Wood, Brian Cole  
Paperback: 1280 pages  
Publisher: O'Reilly Media (November 1, 2002)  
ISBN: 0596004087



Swing, 2e  
by Matthew Robinson, Pavel Vorobiev  
Paperback: 912 pages  
Publisher: Manning (February 2003)  
ISBN: 193011088X



( 金色陽光, 活力Java  
Java 2006 專業技術大會

# Thank you!

侯捷

[jjhou@jjhou.com](mailto:jjhou@jjhou.com)

the  
POWER  
of  
JAVA

