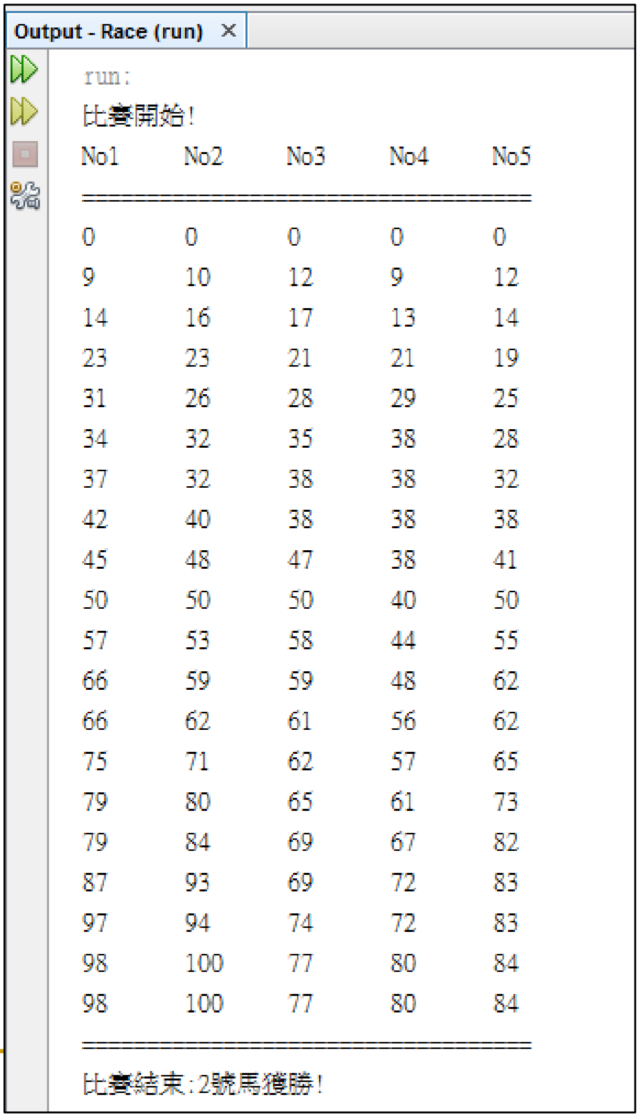
CH19 練習

鄭安翔
ansel_cheng@hotmail.com

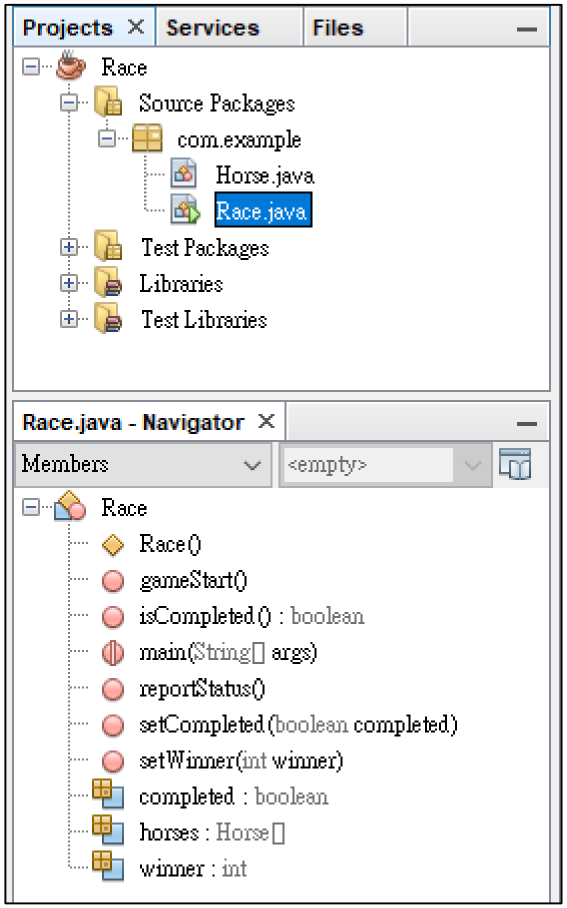
練習1 多執行緒賽馬程式

- 賽馬程式 Race
  - 一場比賽有五匹馬
  - 馬匹每隔100毫秒會隨機前進一段小於10公尺的距離。
  - 跑道總長度為100公尺
  - 當有一匹馬跑到終點時，比賽結束
  - 每隔100毫秒回報馬所在的位置



```
run:
比賽開始!
No1     No2     No3     No4     No5
===================================
0       0       0       0       0
9       10      12      9       12
14      16      17      13      14
23      23      21      21      19
31      26      28      29      25
34      32      35      38      28
37      32      38      38      32
42      40      38      38      38
45      48      47      38      41
50      50      50      40      50
57      53      58      44      55
66      59      59      48      62
66      62      61      56      62
75      71      62      57      65
79      80      65      61      73
79      84      69      67      82
87      93      69      72      83
97      94      74      72      83
98      100     77      80      84
98      100     77      80      84
===================================
比賽結束:2號馬獲勝!
```

Race 類別 / Horse類別

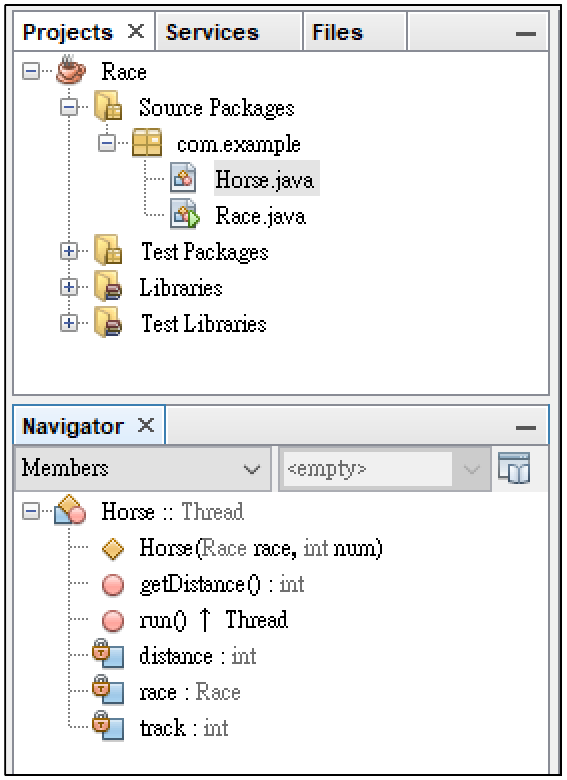

Projects ✕ Services Files

```
Race
  Source Packages
    com.example
      Horse.java
      Race.java
  Test Packages
  Libraries
  Test Libraries
```

Race.java - Navigator ✕

Members <empty>

```
Race
  Race()
  gameStart()
  isCompleted() : boolean
  main(String[] args)
  reportStatus()
  setCompleted(boolean completed)
  setWinner(int winner)
  completed : boolean
  horses : Horse[]
  winner : int
```


Projects ✕ Services Files

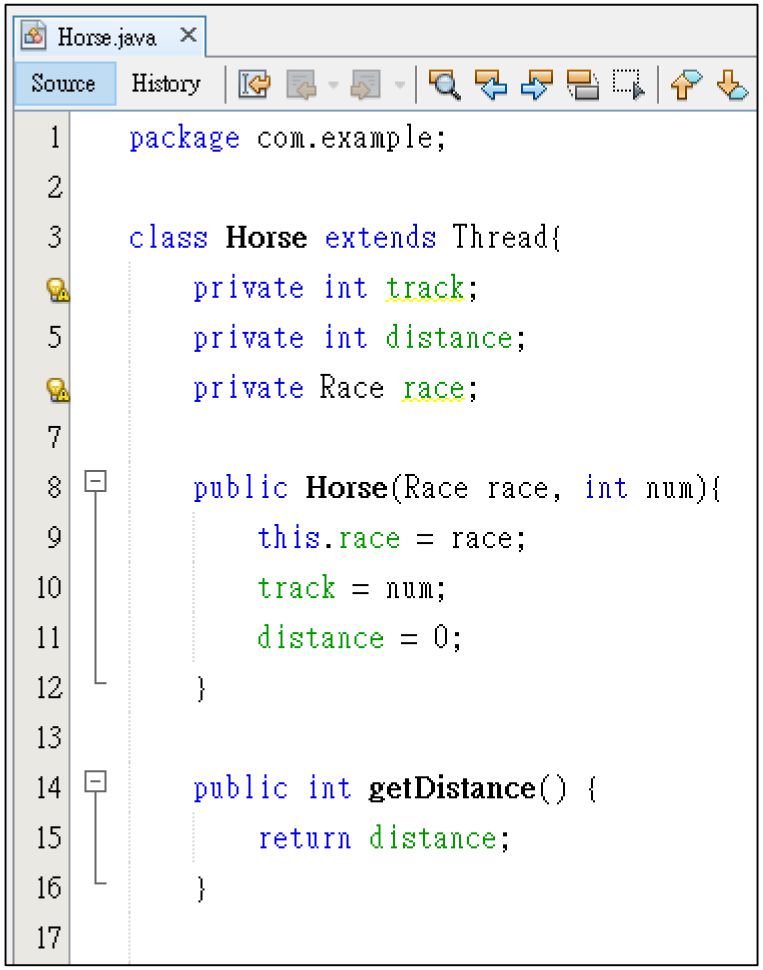
```
Race
  Source Packages
    com.example
      Horse.java
      Race.java
  Test Packages
  Libraries
  Test Libraries
```

Navigator ✕

Members <empty>

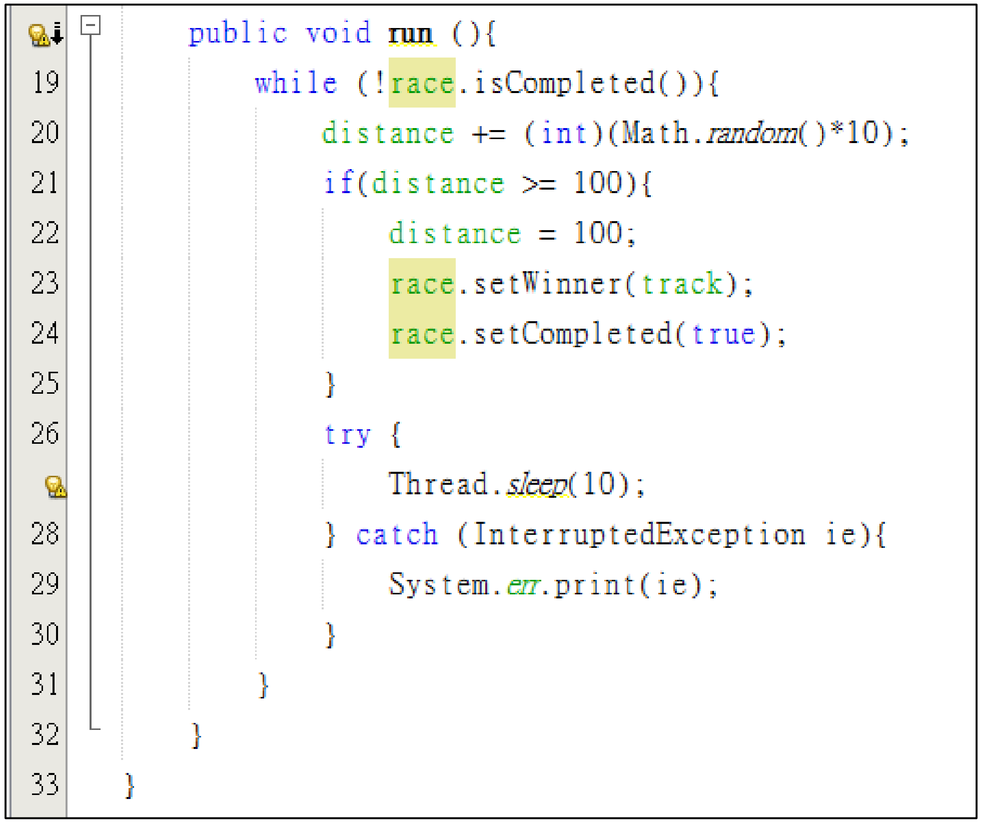
```
Horse :: Thread
  Horse(Race race, int num)
  getDistance() : int
  run() ↑ Thread
  distance : int
  race : Race
  track : int
```

Horse 類別


Horse.java

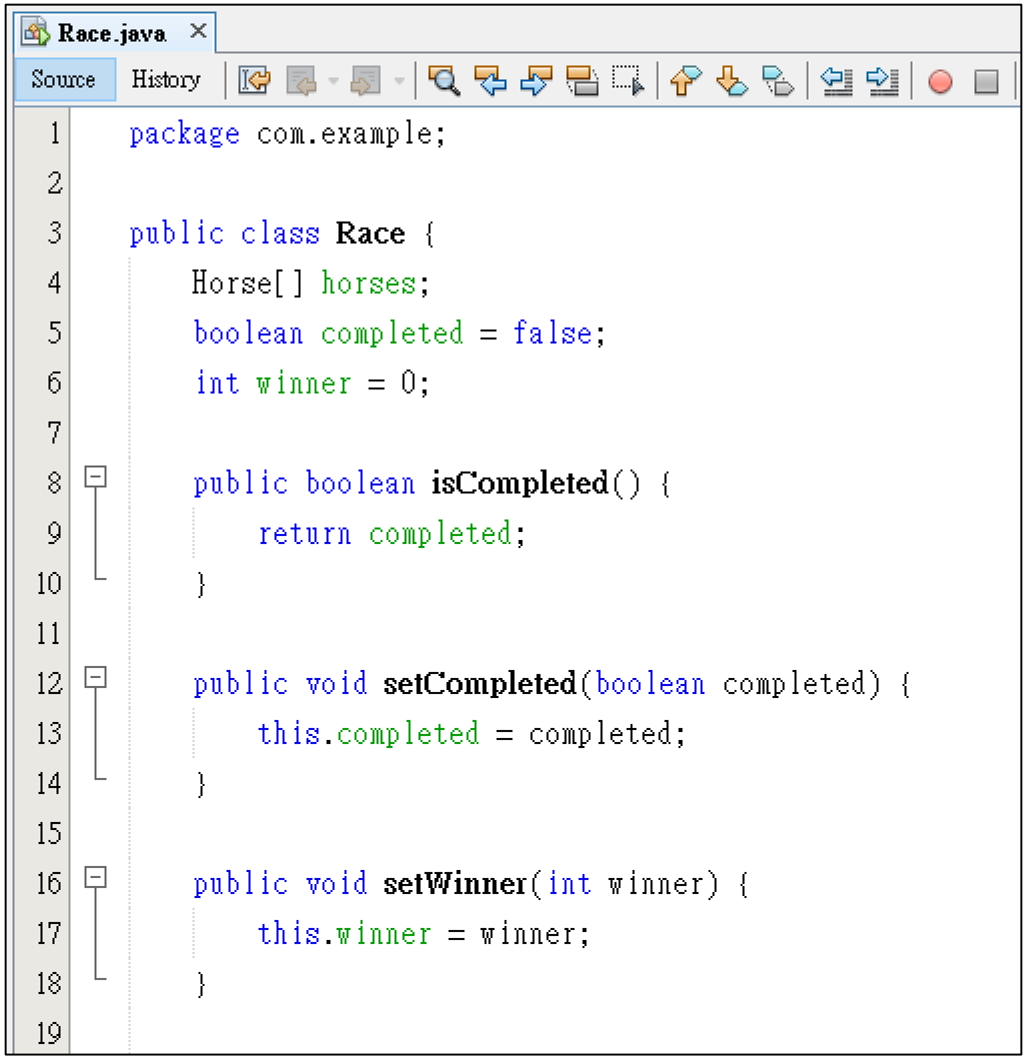
Source History

```
package com.example;

class Horse extends Thread{
    private int track;
    private int distance;
    private Race race;

    public Horse(Race race, int num){
        this.race = race;
        track = num;
        distance = 0;
    }

    public int getDistance() {
        return distance;
    }

```



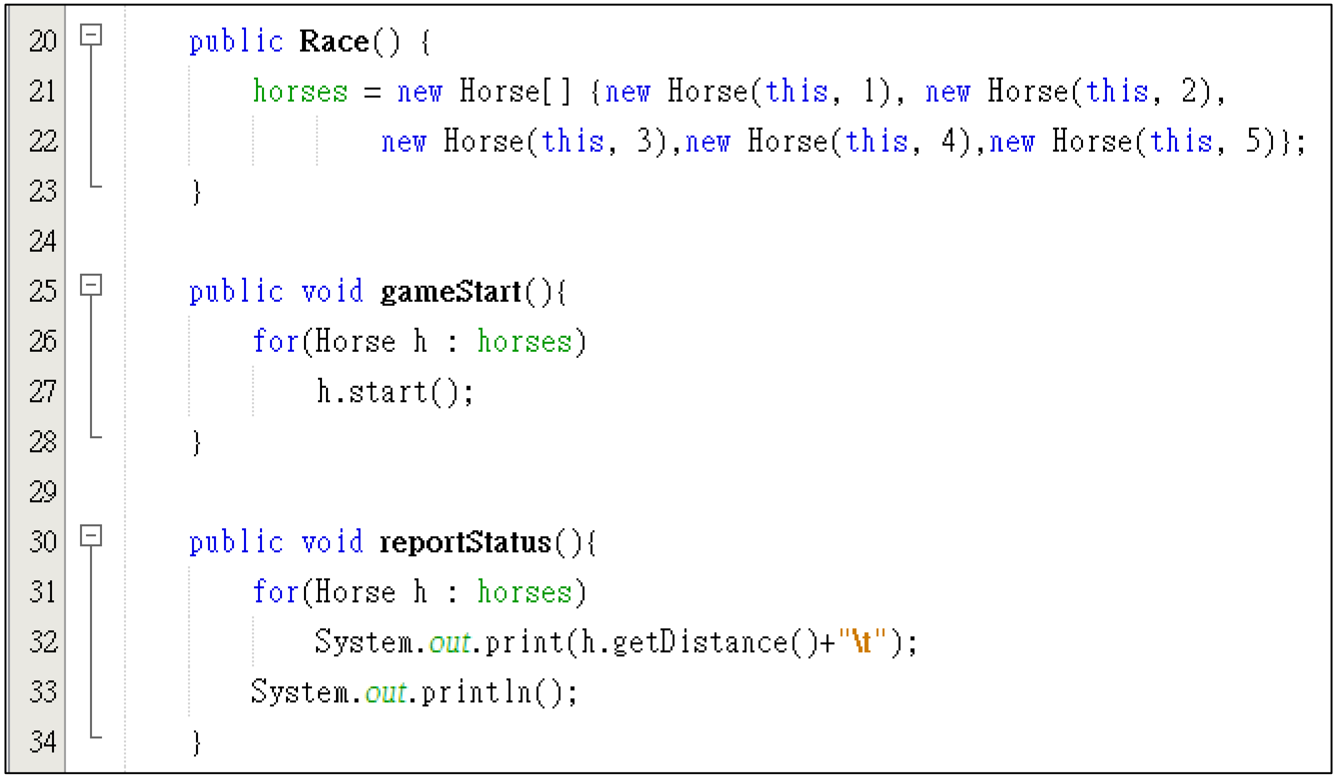
```
    public void run (){
        while (!race.isCompleted()){
            distance += (int)(Math.random()*10);
            if(distance >= 100){
                distance = 100;
                race.setWinner(track);
                race.setCompleted(true);
            }
            try {
                Thread.sleep(10);
            } catch (InterruptedException ie){
                System.err.print(ie);
            }
        }
    }
}
```

Race 類別



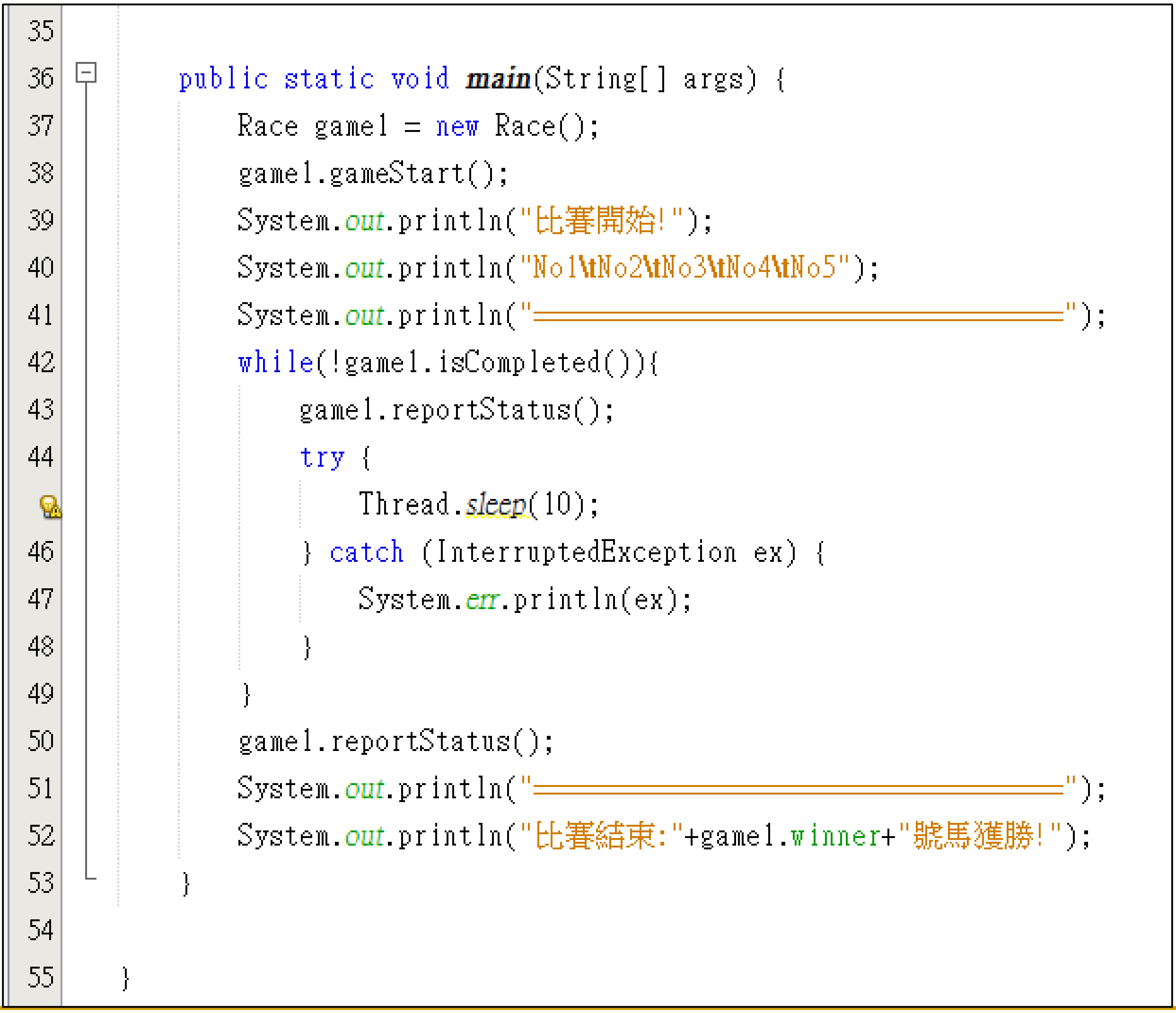
```java
package com.example;

public class Race {
    Horse[] horses;
    boolean completed = false;
    int winner = 0;

    public boolean isCompleted() {
        return completed;
    }

    public void setCompleted(boolean completed) {
        this.completed = completed;
    }

    public void setWinner(int winner) {
        this.winner = winner;
    }

```

Race 類別



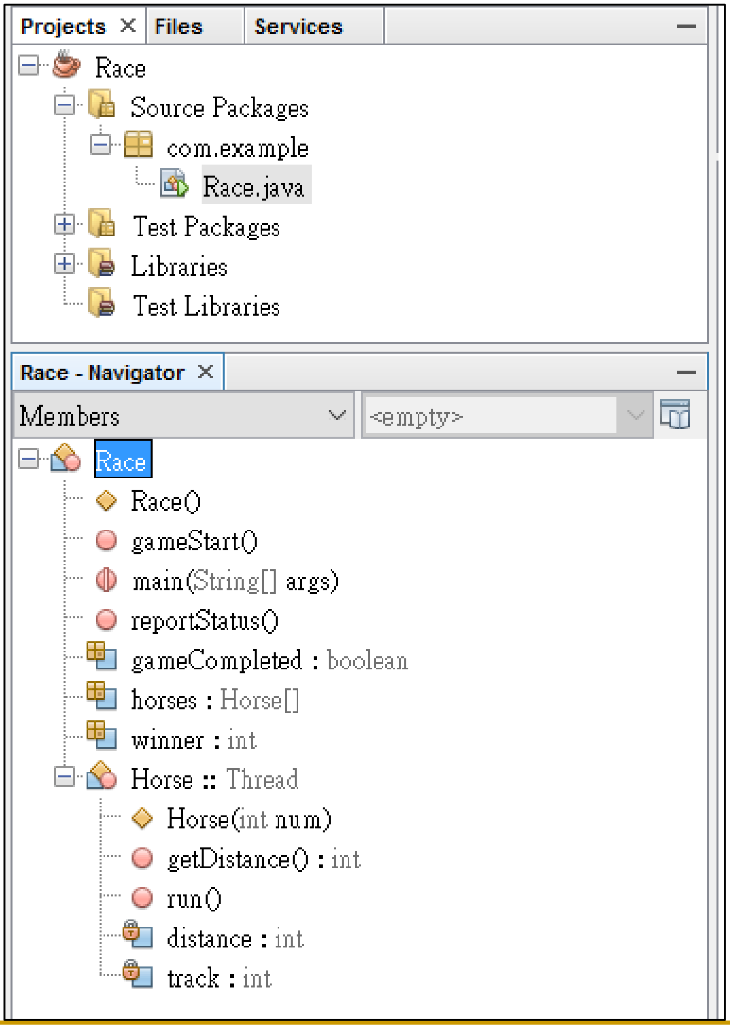
```
    public Race() {
        horses = new Horse[] {new Horse(this, 1), new Horse(this, 2),
                new Horse(this, 3),new Horse(this, 4),new Horse(this, 5)};
    }

    public void gameStart(){
        for(Horse h : horses)
            h.start();
    }

    public void reportStatus(){
        for(Horse h : horses)
            System.out.print(h.getDistance()+"\t");
        System.out.println();
    }
```

Race 類別



```

    public static void main(String[] args) {
        Race game1 = new Race();
        game1.gameStart();
        System.out.println("比賽開始!");
        System.out.println("No1\tNo2\tNo3\tNo4\tNo5");
        System.out.println("=========================================");
        while(!game1.isCompleted()){
            game1.reportStatus();
            try {
                Thread.sleep(10);
            } catch (InterruptedException ex) {
                System.err.println(ex);
            }
        }
        game1.reportStatus();
        System.out.println("=========================================");
        System.out.println("比賽結束:"+game1.winner+"號馬獲勝!");
    }

}
```

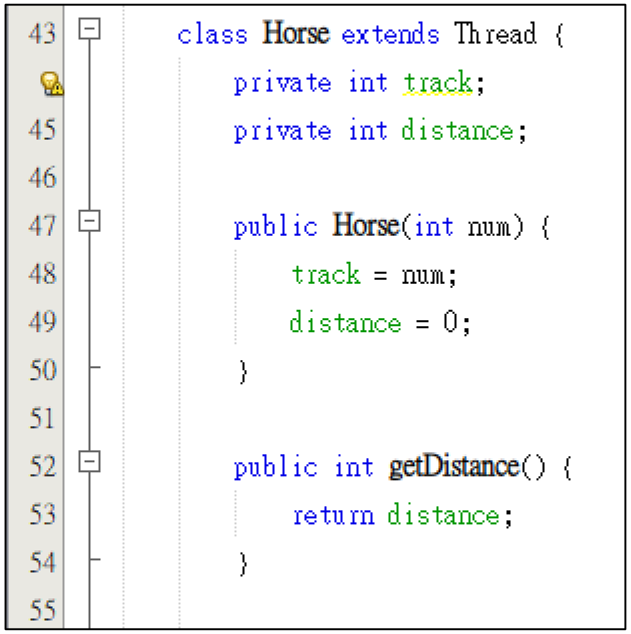
Race 類別 / Race.Horse類別



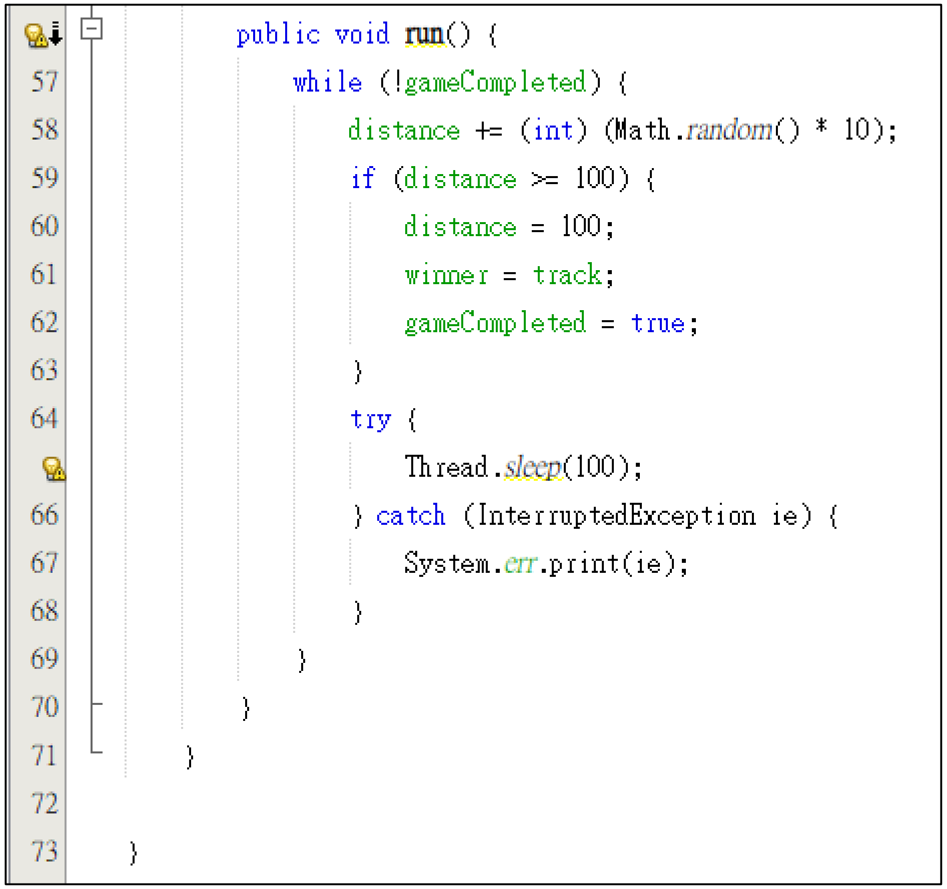
```
Race
  Source Packages
    com.example
      Race.java
  Test Packages
  Libraries
  Test Libraries
```

```
Race
  Race()
  gameStart()
  main(String[] args)
  reportStatus()
  gameCompleted : boolean
  horses : Horse[]
  winner : int
  Horse :: Thread
    Horse(int num)
    getDistance() : int
    run()
    distance : int
    track : int
```

Race.Horse 巢狀類別



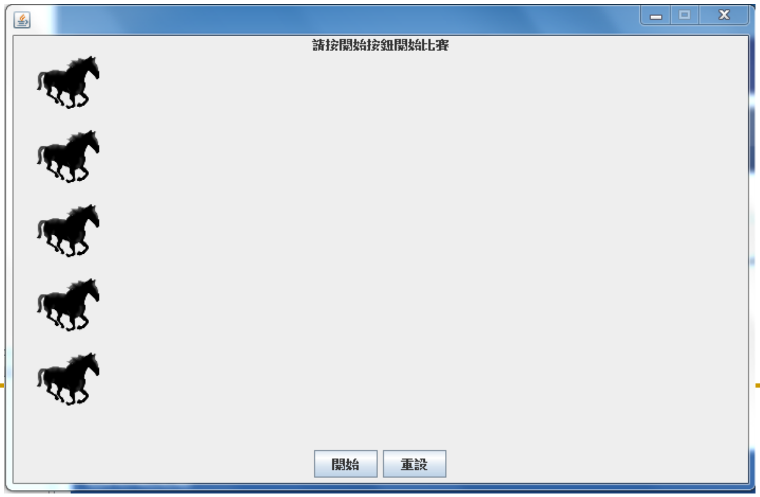
```
class Horse extends Thread {
    private int track;
    private int distance;

    public Horse(int num) {
        track = num;
        distance = 0;
    }

    public int getDistance() {
        return distance;
    }

```

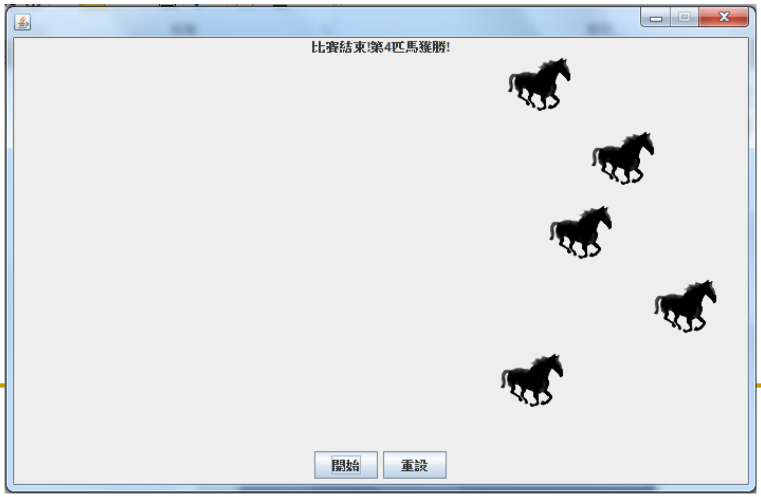


```
    public void run() {
        while (!gameCompleted) {
            distance += (int) (Math.random() * 10);
            if (distance >= 100) {
                distance = 100;
                winner = track;
                gameCompleted = true;
            }
            try {
                Thread.sleep(100);
            } catch (InterruptedException ie) {
                System.err.print(ie);
            }
        }
    }
}

}
```

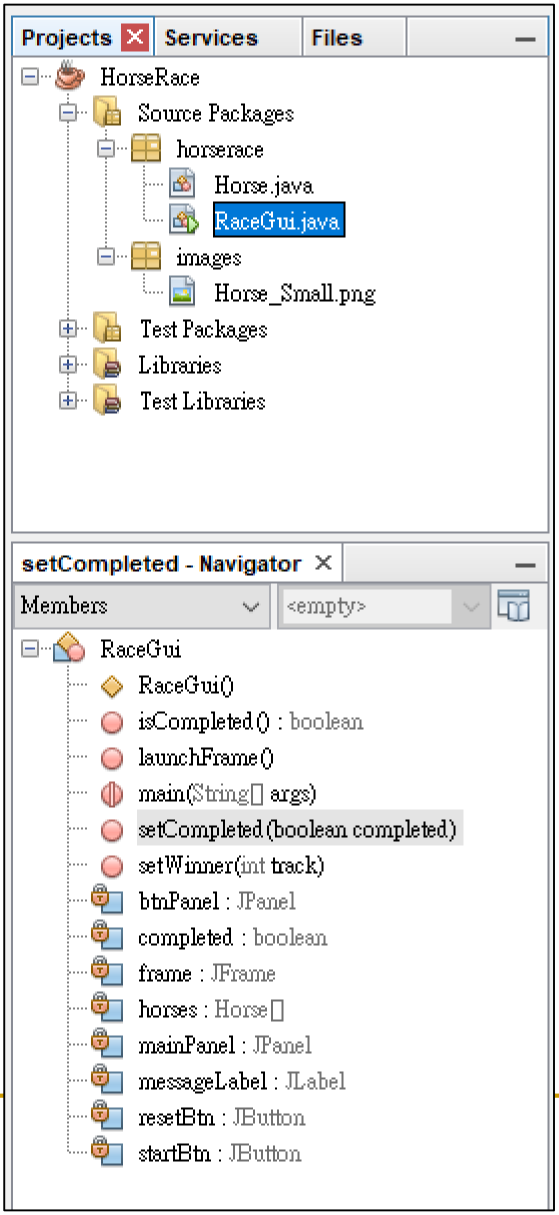
練習1-2 多執行緒賽馬程式

- 賽馬程式 HorseRace
  - 一場比賽有五匹馬，按下開始按紐比賽開始
  - 馬匹每隔100毫秒會隨機前進一段小於50公尺的距離。
  - 跑道總長度為600公尺
  - 當有一匹馬跑到終點時，比賽結束
  - 按下重設按紐清除比賽結果，準備下一場比賽。
  - 執行結果如下圖


請按開始按鈕開始比賽

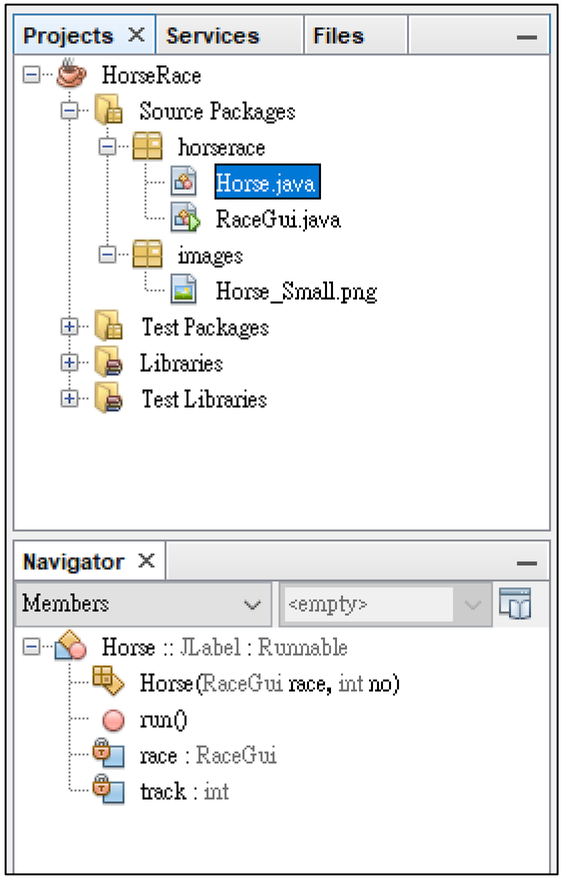

比賽結束!第4匹馬獲勝!

RaceGui 類別



HorseRace
- Source Packages
  - horserace
    - Horse.java
    - RaceGui.java
  - images
    - Horse_Small.png
- Test Packages
- Libraries
- Test Libraries

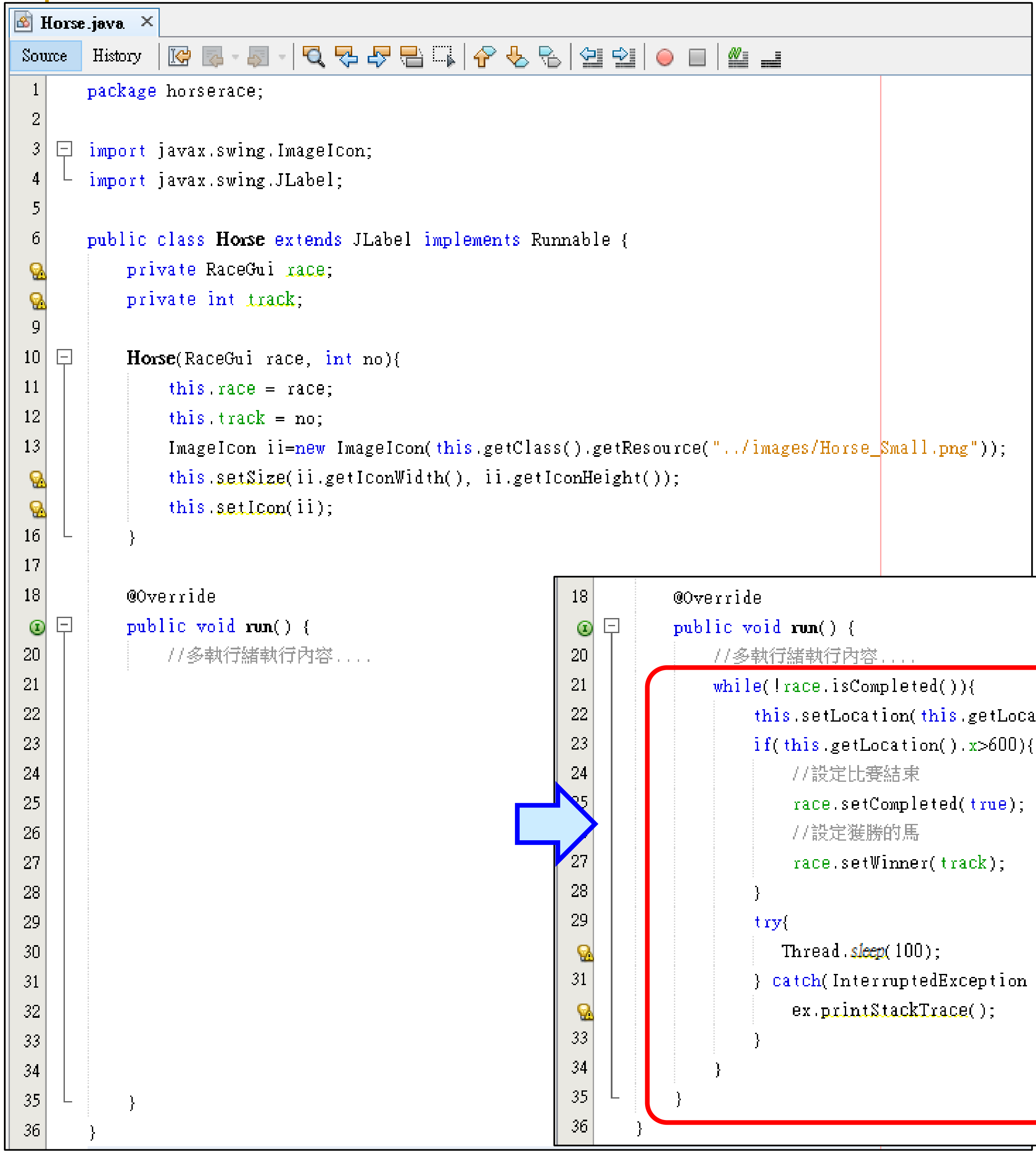
RaceGui
- RaceGui()
- isCompleted() : boolean
- launchFrame()
- main(String[] args)
- setCompleted(boolean completed)
- setWinner(int track)
- btnPanel : JPanel
- completed : boolean
- frame : JFrame
- horses : Horse[]
- mainPanel : JPanel
- messageLabel : JLabel
- resetBtn : JButton
- startBtn : JButton



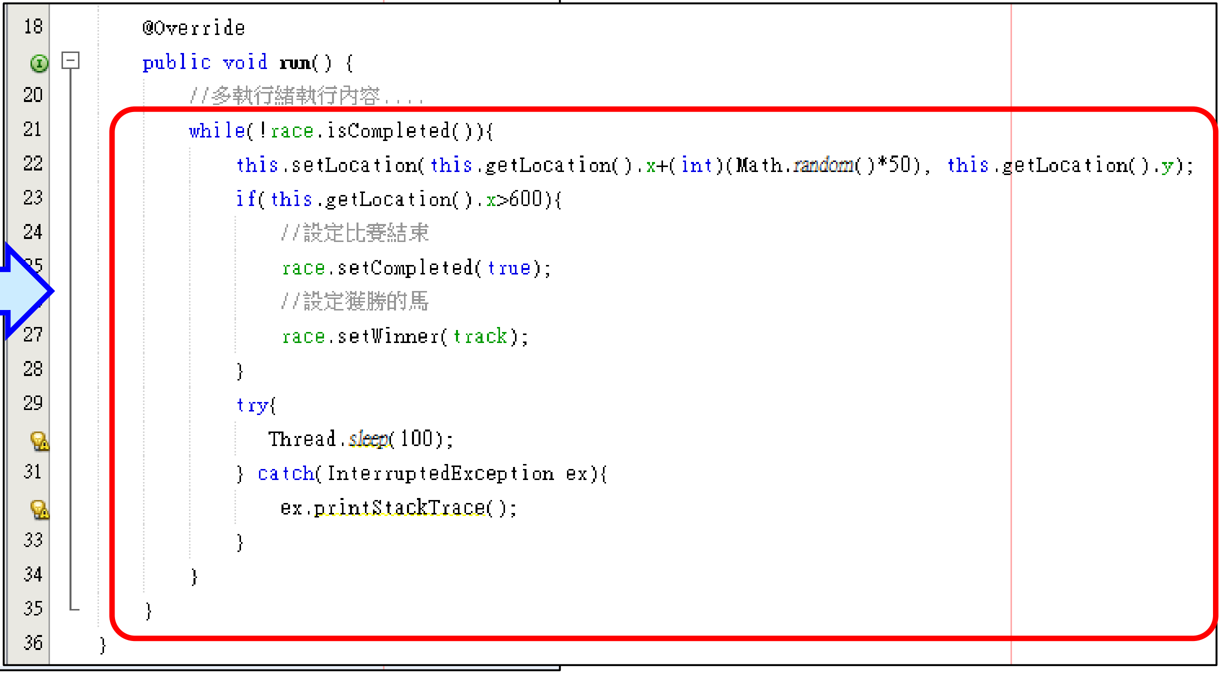
HorseRace
- Source Packages
  - horserace
    - Horse.java
    - RaceGui.java
  - images
    - Horse_Small.png
- Test Packages
- Libraries
- Test Libraries

Horse :: JLabel : Runnable
- Horse(RaceGui race, int no)
- run()
- race : RaceGui
- track : int

Horse 類別

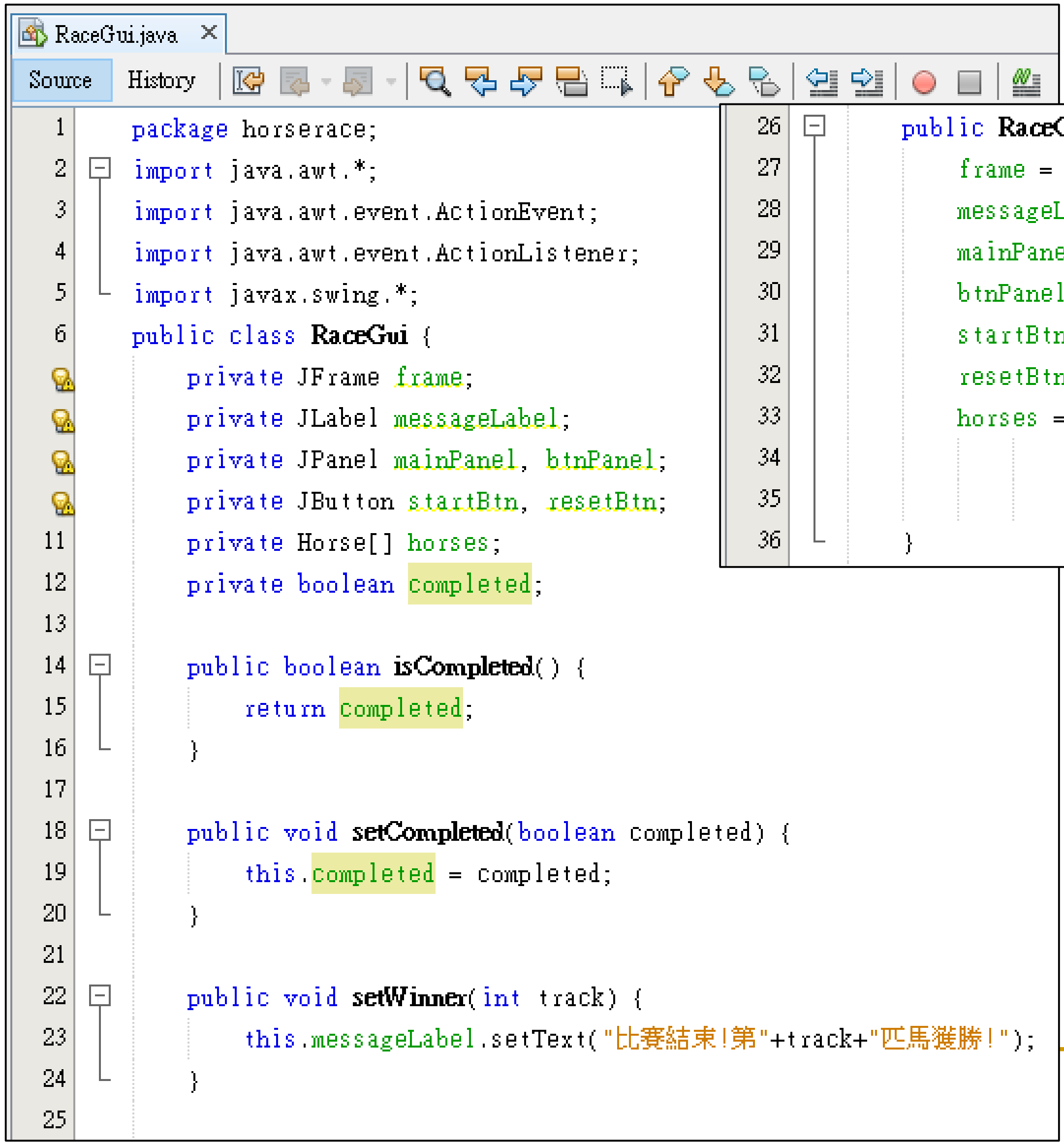


```
package horserace;

import javax.swing.ImageIcon;
import javax.swing.JLabel;

public class Horse extends JLabel implements Runnable {
    private RaceGui race;
    private int track;

    Horse(RaceGui race, int no){
        this.race = race;
        this.track = no;
        ImageIcon ii=new ImageIcon(this.getClass().getResource("../images/Horse_Small.png"));
        this.setSize(ii.getIconWidth(), ii.getIconHeight());
        this.setIcon(ii);
    }

    @Override
    public void run() {
        //多執行緒執行內容....














    }
}
```

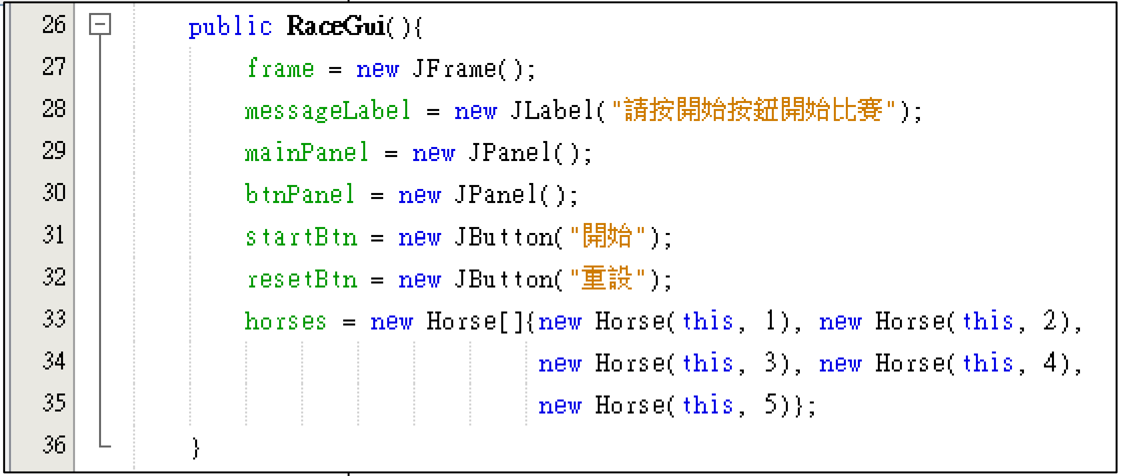


```
    @Override
    public void run() {
        //多執行緒執行內容....
        while(!race.isCompleted()){
            this.setLocation(this.getLocation().x+(int)(Math.random()*50), this.getLocation().y);
            if(this.getLocation().x>600){
                //設定比賽結束
                race.setCompleted(true);
                //設定獲勝的馬
                race.setWinner(track);
            }
            try{
                Thread.sleep(100);
            } catch(InterruptedException ex){
                ex.printStackTrace();
            }
        }
    }
}
```

RaceGui 類別

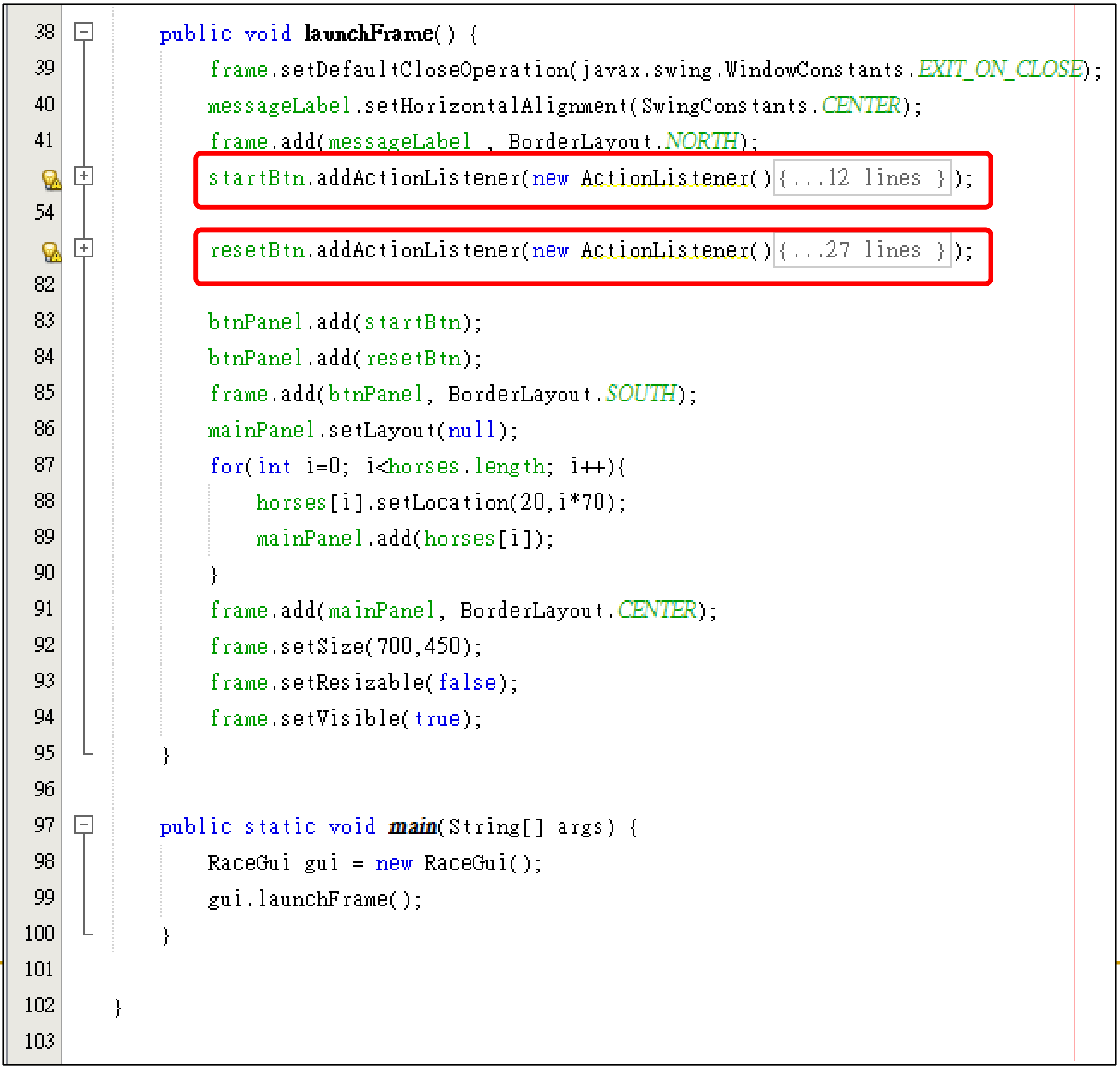


```java
package horserace;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.*;
public class RaceGui {
    private JFrame frame;
    private JLabel messageLabel;
    private JPanel mainPanel, btnPanel;
    private JButton startBtn, resetBtn;
    private Horse[] horses;
    private boolean completed;

    public boolean isCompleted() {
        return completed;
    }

    public void setCompleted(boolean completed) {
        this.completed = completed;
    }

    public void setWinner(int track) {
        this.messageLabel.setText("比賽結束!第"+track+"匹馬獲勝!");
    }

```



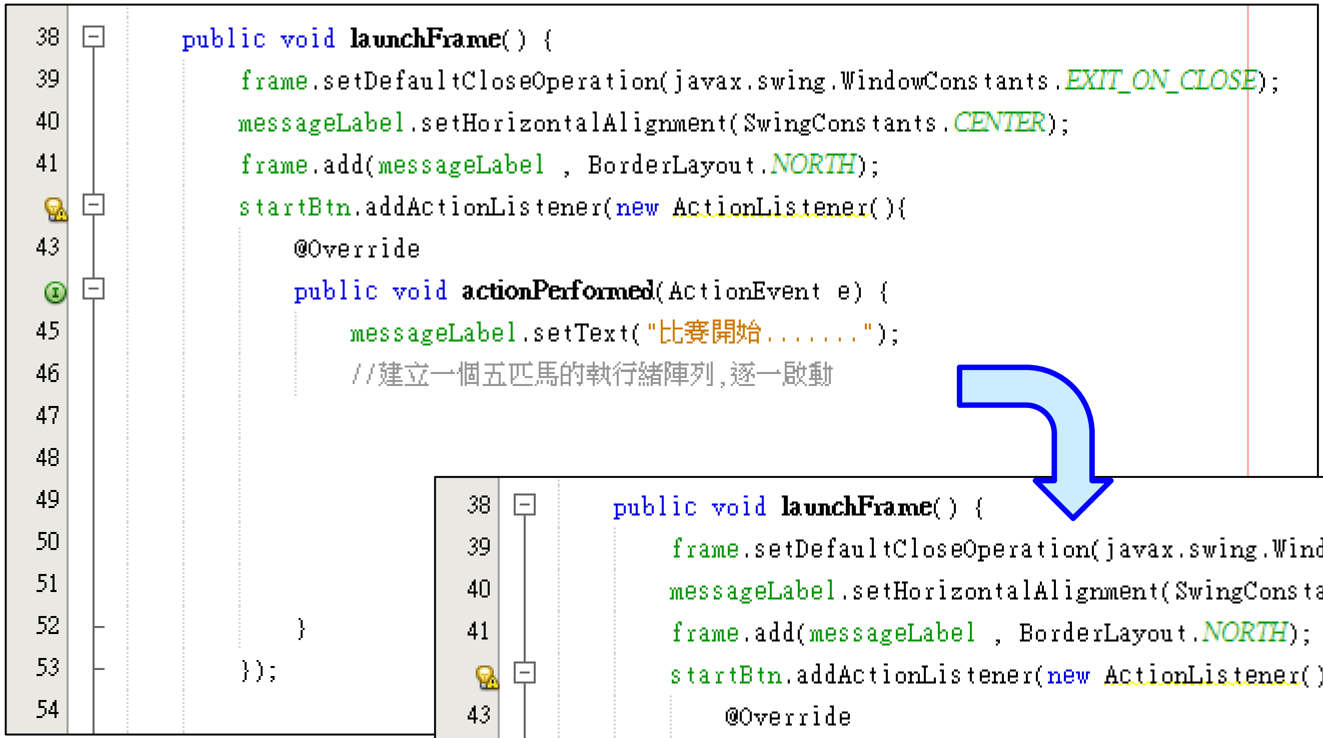
```java
    public RaceGui(){
        frame = new JFrame();
        messageLabel = new JLabel("請按開始按鈕開始比賽");
        mainPanel = new JPanel();
        btnPanel = new JPanel();
        startBtn = new JButton("開始");
        resetBtn = new JButton("重設");
        horses = new Horse[]{new Horse(this, 1), new Horse(this, 2),
                             new Horse(this, 3), new Horse(this, 4),
                             new Horse(this, 5)};
    }
```

RaceGui 類別

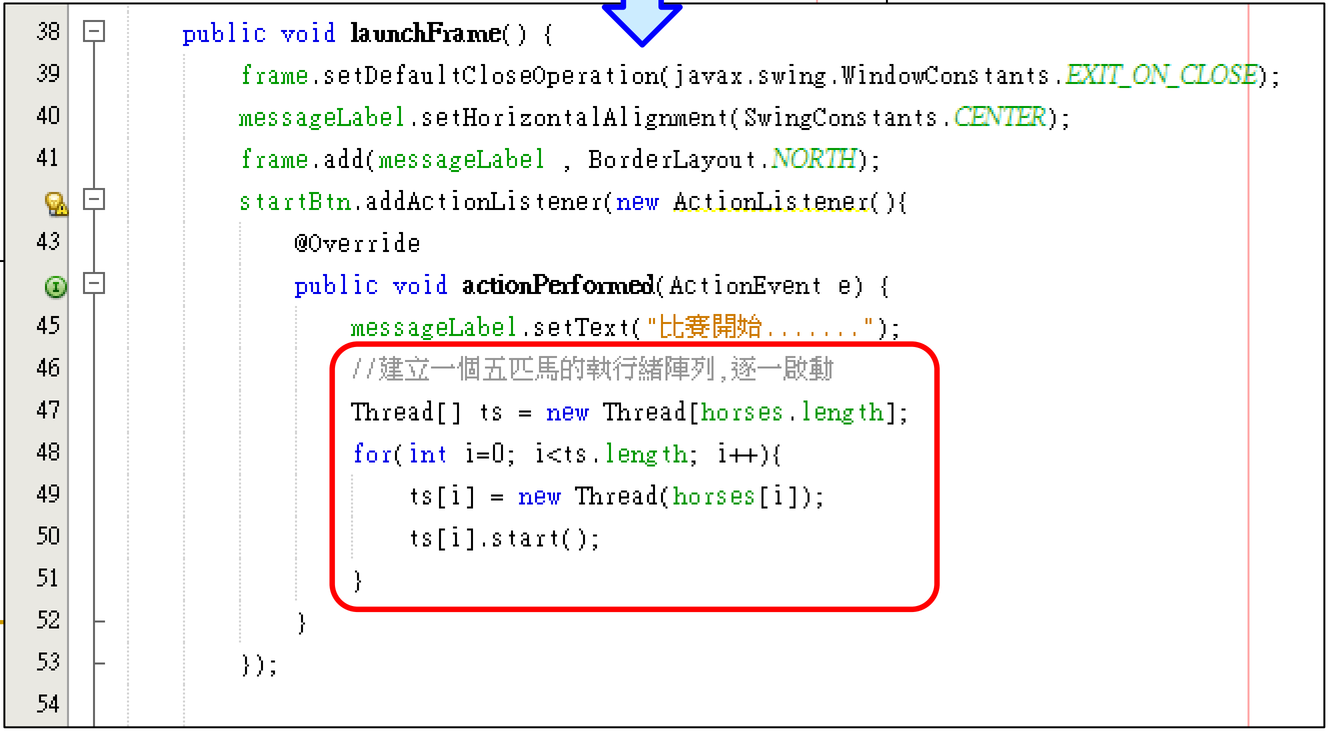


```
    public void launchFrame() {
        frame.setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        messageLabel.setHorizontalAlignment(SwingConstants.CENTER);
        frame.add(messageLabel , BorderLayout.NORTH);
        startBtn.addActionListener(new ActionListener(){...12 lines });

        resetBtn.addActionListener(new ActionListener(){...27 lines });

        btnPanel.add(startBtn);
        btnPanel.add( resetBtn);
        frame.add(btnPanel, BorderLayout.SOUTH);
        mainPanel.setLayout(null);
        for(int i=0; i<horses.length; i++){
            horses[i].setLocation(20,i*70);
            mainPanel.add(horses[i]);
        }
        frame.add(mainPanel, BorderLayout.CENTER);
        frame.setSize(700,450);
        frame.setResizable(false);
        frame.setVisible(true);
    }

    public static void main(String[] args) {
        RaceGui gui = new RaceGui();
        gui.launchFrame();
    }

}
```

RaceGui 類別

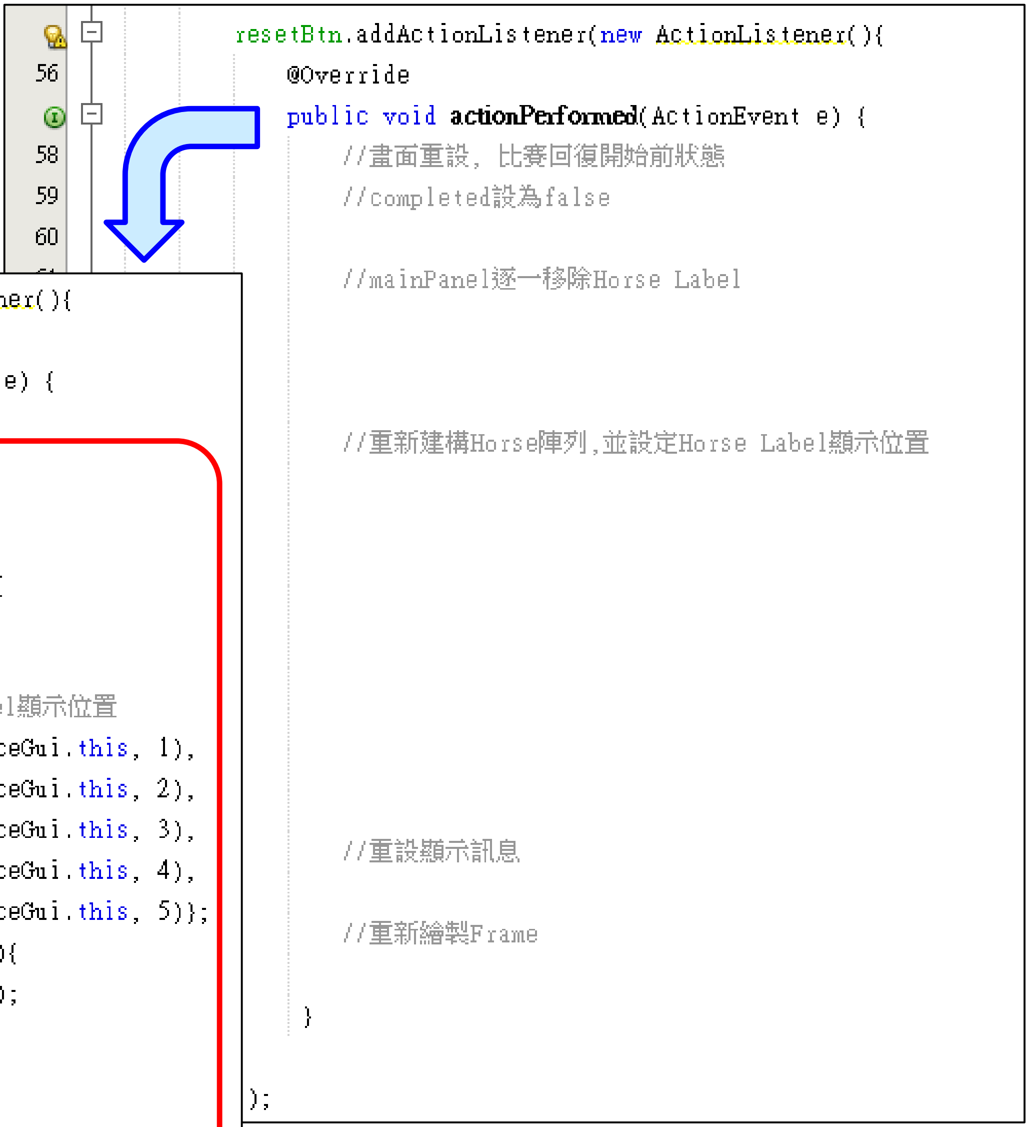


```
    public void launchFrame() {
        frame.setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        messageLabel.setHorizontalAlignment(SwingConstants.CENTER);
        frame.add(messageLabel , BorderLayout.NORTH);
        startBtn.addActionListener(new ActionListener(){
            @Override
            public void actionPerformed(ActionEvent e) {
                messageLabel.setText("比賽開始.......");
                //建立一個五匹馬的執行緒陣列,逐一啟動




            }
        });
```

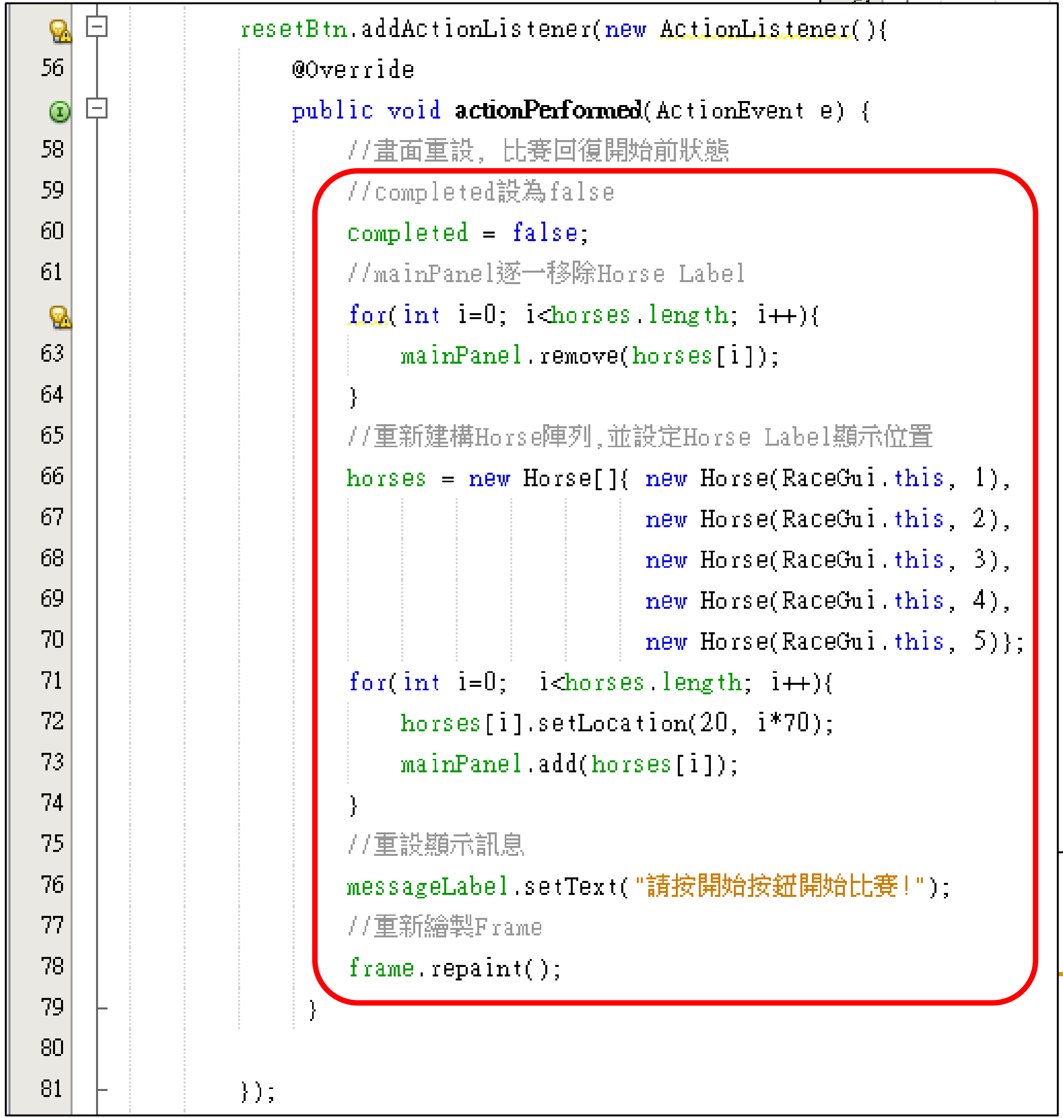


```
    public void launchFrame() {
        frame.setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        messageLabel.setHorizontalAlignment(SwingConstants.CENTER);
        frame.add(messageLabel , BorderLayout.NORTH);
        startBtn.addActionListener(new ActionListener(){
            @Override
            public void actionPerformed(ActionEvent e) {
                messageLabel.setText("比賽開始.......");
                //建立一個五匹馬的執行緒陣列,逐一啟動
                Thread[] ts = new Thread[horses.length];
                for(int i=0; i<ts.length; i++){
                    ts[i] = new Thread(horses[i]);
                    ts[i].start();
                }
            }
        });
```

RaceGui 類別



```
        resetBtn.addActionListener(new ActionListener(){
            @Override
            public void actionPerformed(ActionEvent e) {
                //畫面重設，比賽回復開始前狀態
                //completed設為false

                //mainPanel逐一移除Horse Label



                //重新建構Horse陣列,並設定Horse Label顯示位置









                //重設顯示訊息

                //重新繪製Frame

            }

        });
```



```
        resetBtn.addActionListener(new ActionListener(){
            @Override
            public void actionPerformed(ActionEvent e) {
                //畫面重設，比賽回復開始前狀態
                //completed設為false
                completed = false;
                //mainPanel逐一移除Horse Label
                for(int i=0; i<horses.length; i++){
                    mainPanel.remove(horses[i]);
                }
                //重新建構Horse陣列,並設定Horse Label顯示位置
                horses = new Horse[]{ new Horse(RaceGui.this, 1),
                                      new Horse(RaceGui.this, 2),
                                      new Horse(RaceGui.this, 3),
                                      new Horse(RaceGui.this, 4),
                                      new Horse(RaceGui.this, 5)};
                for(int i=0;  i<horses.length; i++){
                    horses[i].setLocation(20, i*70);
                    mainPanel.add(horses[i]);
                }
                //重設顯示訊息
                messageLabel.setText("請按開始按鈕開始比賽!");
                //重新繪製Frame
                frame.repaint();
            }

        });
```