Rapport of MOOC S3

du 23/09 2019 au 8/12 2019

Par Deng yufeng

dyufeng1162@gmail.com

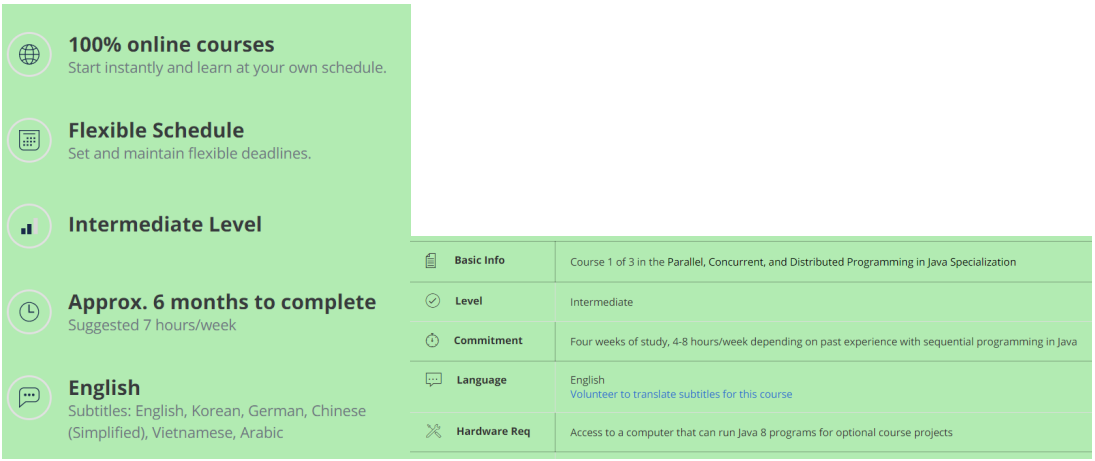
Tuteur: ROUSSEL David

Overview

This repport is the synthesis of my MOOC course in the coursera about "Object Oriented Programming, Parallel, Distributed Programming".it starts from 22/09 to 08/12.

Totally,i have learned something about Object Oriented Programming in the last semester,but i haven't understand it clearly owing to less of fondement informatic,and i want to control a programming language especially java more detailly, So this semester, i choose the course once again.

There are two part in my chosen courses in this specialization,one of which is about Object Oriented Programming in Java(San Diego University) more fundamentally.The other one is about Parallel, Concurrent, and Distributed Programming in Java Specialization(Rice University).



The first course consists of 4 parts.Due to previous course in ENSIIE, i spend about two weeks to reviewing the knowledge quicklly,and then 6 weeks to finish the second part. The second course , i start studying Parallel part for two weeks.

Because space is limited, I choose some content statements.

I take the two course of them, i have finished 10 weeks(about 50+ hours) until 8/12,2019, beacuse the course is audit , attend (a class) informally, without working for credit,if not paid. Every week,there is a small quizz with 8 choice questions at last,it can't verify it,but i finished it and verified it by internet,

Contents

Introduction

1. Presentation

The first course :Object Oriented Programming in Java

This course is designed as a three-part series and covers a theme or body of knowledge through various video lectures, demonstrations, and coding projects. Each lecture is followed by summary text to help to review the concepts covered in the video, along with links for optional reading. (the optional reading to freely available material such as WIkipedia articles and Java tutorials, rather than textbooks.) At the end of the lecture series for a module, there will be a short 10-question quiz.

Because space is limited, I choose some main content statements.

The main task:Building your own interactive world map

In order to build an interative world map,I will do what professional developers do all the time and work with existing libraries.

In particular, by using the UnfoldingMaps library and Processing library. Generate visual maps using Java code and the Unfolding Maps libraries.

There are 6 modules.The first two weeks is to famaliar two libraries and module 6 contains the final output of the program.

Some details for every week:

week3

> - Create and manipulate Graphical User Interfaces (GUIs)
> - Find and effectively use Java documentation
> - Use the Processing library in your programs
> - Customize visual maps using Java code
> - Use Java to read and parse data files
> - Organize data into appropriate objects
> - Describe and use the List data type and the ArrayList class
> - Describe and use the Map data type

week 4

> - Explain the value of inheritance as a feature in object oriented programming languages
> - Use the keyword extends
> - Explain the relationship between a superclass and a subclass
> - Use UML Diagrams to display class hierarchies
> - Explain an “is-a” relationship between classes
> - Describe how Java object construction occurs from the inside out
> - Use same-class and superclass constructors in class creation
> - Create methods which override from a superclass
> - Contrast method overloading and method overriding
> - Explain the purpose of polymorphism
> - Step through decisions made at compile time and runtime

- Use casting of objects to aid the compiler
- Use the keyword "abstract"
- Compare and contrast “inheritance of implementation” and “inheritance of interface”
- Decide between Abstract Classes and Interfaces

week 5

- Distinguish between procedural programming and event-driven programming. Trace event-driven code
- Explain how mouse-clicks and keyboard input interact with the program execution
- Design user-interface elements
- Write code to implement a button and respond to mouse events
- Describe inheritance relationships used in the course so far
- Distinguish between interfaces and classes in this hierarchy

week 6

- Motivate the importance of search
- Write code to perform a linear search
- Explain and implement the binary search algorithm
- Explain why binary search is better than linear search
- Define sorting
- Explain why sorting data can be useful
- Explain and implement the selection sort algorithm
- Trace code and describe its high-level function
- Describe alternate algorithms for sorting
- Use pre-defined Java methods to sort
- Explain the properties of this built-in sort algorithm
- Define and use the Comparable interface in Java

The second course: The fork/join framework

At the beginning,it introduces that the fork/join framework was presented in Java 7. It provides tools to help speed up parallel processing by attempting to use all available processor cores – which is accomplished through a divide and conquer approach. Some key differences between future tasks and regular tasks in the FJ framework and RecursiveTask class and RecursiveAction class in the FJ framework.

More details follow.

2. Summary of some points(knowledge presented in the MOOC)

Here , i will present some diffcult or important points that i have learned from this course.

explain how mouse-clicks and keyboard input interact with the program execution

Search for data

Linear Search Algorithm

```
public static String findAirportCode(String toFind,Airport[] airports)
{
  for(int i=0;i<=airports.length();i++)
  {
    if(airports[i].getString.equals(toFind))
      return toFind;

  }
  retrun null;
}
```

Selection sort

Sorting data means arranging it in a certain order, often in an array-like data structure

There are various sorting algorithms.In this course , it introduces Bubble Sort,Merge Sort,Sorting in Java.

Bubble sort:Bubble sort works by swapping adjacent elements if they're not in the desired order. This process repeats from the beginning of the array until all elements are in order.

Here are the steps for sorting an array of numbers from least to greatest:

- 4 2 1 5 3: The first two elements are in the wrong order, so we swap them.
- 2 4 1 5 3: The second two elements are in the wrong order too, so we swap.
- 2 1 4 5 3: These two are in the right order, 4 < 5, so we leave them alone.
- 2 1 4 5 3: Another swap.

- 2 1 4 3 5: Here's the resulting array after one iteration.

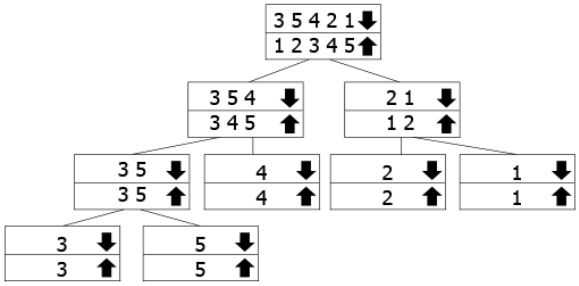
Because at least one swap occurred during the first pass (there were actually three), we need to go through the whole array again and repeat the same process. By repeating this process, until no more swaps are made, we'll have a sorted array.

Merge Sort:Merge sort uses recursion to solve the problem of sorting more efficiently than algorithms previously presented, and in particular it uses a divide and conquer approach. Using both of these concepts, we'll break the whole array down into two subarrays and then:

- 1.Sort the left half of the array (recursively)
- 2.Sort the right half of the array (recursively)
- 3.Merge the solutions

This tree is meant to represent how the recursive calls work. The arrays marked with the down arrow are the ones we call the function for, while we're merging the up arrow ones going back up. So you follow the down arrow to the bottom of the tree, and then go back up and merge.

In our example, we have the array 3 5 4 2 1, so we divide it into 3 5 4 and 2 1. To sort them, we further divide them into their components. Once we've reached the bottom, we start merging up and sorting them as we go.



Code implementation:

```
void merge(int[] array, int left, int mid, int right) {
   // calculating lengths
   int lengthLeft = mid - left + 1;
   int lengthRight = right - mid;

   // creating temporary subarrays
   int leftArray[] = new int [lengthLeft];
   int rightArray[] = new int [lengthRight];

   // copying our sorted subarrays into temporaries
   for (int i = 0; i < lengthLeft; i++)
       leftArray[i] = array[left+i];
   for (int i = 0; i < lengthRight; i++)
       rightArray[i] = array[mid+i+1];

   // iterators containing current index of temp subarrays
   int leftIndex = 0;
```

```
    int rightIndex = 0;

    // copying from leftArray and rightArray back into array
    for (int i = left; i < right + 1; i++) {
        // if there are still uncopied elements in R and L, copy minimum of the
two
        if (leftIndex < lengthLeft && rightIndex < lengthRight) {
            if (leftArray[leftIndex] < rightArray[rightIndex]) {
                array[i] = leftArray[leftIndex];
                leftIndex++;
            }
            else {
                array[i] = rightArray[rightIndex];
                rightIndex++;
            }
        }
        // if all the elements have been copied from rightArray, copy the rest of
leftArray
        else if (leftIndex < lengthLeft) {
            array[i] = leftArray[leftIndex];
            leftIndex++;
        }
        // if all the elements have been copied from leftArray, copy the rest of
rightArray
        else if (rightIndex < lengthRight) {
            array[i] = rightArray[rightIndex];
            rightIndex++;
        }
    }
}
```

Sorting with Comparable and Comparator in Java

Comparable Interface

In the final piece of my project, i will organize the earthquake data and compute statistics on it. ,it sorts earthquakes in reverse order of magnitude. so i need to sort elements from a database into a collection, array, or map. beacuse i need sort by earthquake's magnitude and name of place, Using the Comparable interface and compareTo() method.

In Java, we can implement whatever sorting algorithm we want with any type. we can sort using alphabetical order, String length, reverse alphabetical order, or numbers.

To do this, my class must implement the Comparable interface, where T is the type, and override a method called .compareTo(). This method returns a negative integer if this is smaller than the argument element, 0 if they're equal, and a positive integer if this is greater.

For example, we've made a class Student, and each student is identified by an id and a year they started their studies. We want to sort them primarily by generations, but also secondarily by IDs:

```
public static class Student implements Comparable<Student> {
    int studentId;
    int studentGeneration;

    public Student(int studentId, int studentGeneration) {
        this.studentId = studentId;
        this.studentGeneration = studentGeneration;
    }

    @Override
    public String toString() {
        return studentId + "/" + studentGeneration % 100;
    }

    @Override
    public int compareTo(Student student) {
        int result = this.studentGeneration - student.studentGeneration;
        if (result != 0)
            return result;
        else
            return this.studentId - student.studentId;
    }
}
```

And here's how to use it in an application:

```
public static void main(String[] args) {
    Student[] a = new SortingAlgorithms.Student[5];
    a[0] = new Student(75, 2016);
    a[1] = new Student(52, 2019);
    a[2] = new Student(57, 2016);
    a[3] = new Student(220, 2014);
    a[4] = new Student(16, 2018);

    Arrays.sort(a);

    System.out.println(Arrays.toString(a));
}
```

Output:

[220/14, 57/16, 75/16, 16/18, 52/19]

The **compareTo()** method compares a given object or the current instance with a specified object to determine the order of objects. Here's a quick look at how compareTo() works:

If the comparison returns	Then ...
>= 1	this.Id > student.Id

If the comparison returns	Then ...
0	this.Id == student.Id
<= -1	this.Id < student.Id

We can only use classes that are comparable with the sort() method. If we try to pass a class that does not implement Comparable, we will receive a compilation error.

What occurs when object is created in Java?

when a new object is created, as long as no exceptions occur:

> - 1.Memory is allocated.
> - 2.Fields are initialized to their default values.
> - 3.The "first line" of the chosen constructor is invoked, unless it's an Object. By first line I mean either explicit call to super() or this(), or an implicit call to super().
> - 4.The instance initializer is executed and the fields are initialized to their requested values (actually field initialization is usually compiled as an inline part of the instance initializer).
> - 5.The rest of the constructor code is executed.

So how Java object construction occurs from the inside out

Constructor chaining is the process of calling one constructor from another constructor with respect to current object. Constructor chaining can be done in two ways:

> - **Within same class**: It can be done using **this()** keyword for constructors in same class
> - **From base class**: by using **super()** keyword to call constructor from the base class.

Constructor chaining occurs through inheritance. A sub class constructor's task is to call super class's constructor first. This ensures that creation of sub class's object starts with the initialization of the data members of the super class. There could be any numbers of classes in inheritance chain. Every constructor calls up the chain till class at the top is reached.

Why do we need constructor chaining ?

This process is used when we want to perform multiple tasks in a single constructor rather than creating a code for each task in a single constructor we create a separate constructor for each task and make their chain which makes the program more readable.

Contrast method overloading and method overriding

Definitions

Overloading occurs when two or more methods in one class have the same method name but different parameters.
Overriding means having two methods with the same method name and parameters (i.e., method signature). One of the methods is in the parent class and the other is in the child class. Overriding allows a child class to provide a specific implementation of a method that is already provided its parent class.

> - 1). The real object type in the run-time, not the reference variable's type, determines which overridden method is used at runtime. In contrast, reference type determines which overloaded method will be used at compile time.
> - 2). Polymorphism applies to overriding, not to overloading.
> - 3). Overriding is a run-time concept while overloading is a compile-time concept.

Difference between Runtime Polymorphism and Compile time Polymorphism

Compile time Polymorphism (or Static polymorphism)

Polymorphism that is resolved during compiler time is known as static polymorphism. Method overloading is an example of compile time polymorphism.

Runtime Polymorphism (or Dynamic polymorphism)

It is also known as Dynamic Method Dispatch. Dynamic polymorphism is a process in which a call to an overridden method is resolved at runtime, thats why it is called runtime polymorphism

Procedure-driven and Event-driven Programming

Procedure-driven programming, which we can think of as traditional programming, defines the programming process as the development of procedures that explicitly direct the flow of data and control.

Event-driven programming defines the programming process as the development of procedures that respond to the flow of data and control as directed by the user, program, or operating system.

Explain how mouse-clicks and keyboard input interact with the program execution

The physical mouse object is used to control the position of the cursor on screen and to select interface elements. The cursor position is read by computer programs as two numbers, the x-coordinate and the y-coordinate. These numbers can be used to control attributes of elements on screen.

If these coordinates are collected and analyzed, they can be used to extract higher-level information such as the speed and direction of the mouse. This data can in turn be used for gesture and pattern recognition.

Keyboards are typically used to input characters for composing documents, email, and instant message.

We can use these information by Processing library

Mouse Data

The Processing variables mouseX and mouseY (note the capital X and Y) store the x-coordinate and y-coordinate of the cursor relative to the origin in the upper-left corner of the display window.
To see the actual values produced while moving the mouse, run this program to print the values to the console:

```
void draw() {
  frameRate(12);
  println(mouseX + " : " + mouseY);
}
```

Mouse buttons

Computer mice and other related input devices typically have between one and three buttons; Processing can detect when these buttons are pressed with the mousePressed and mouseButton variables. Used with the button status, the cursor position enables the mouse to perform different actions.

For example, a button press when the mouse is over an icon can select it, so the icon can be moved to a different location on screen. The mousePressed variable is true if any mouse button is pressed and false if no mouse button is pressed. The variable mouseButton is LEFT, CENTER, or RIGHT depending on the mouse button most recently pressed.

The mousePressed variable reverts to false as soon as the button is released, but the mouseButton variable retains its value until a different button is pressed.

These variables can be used independently or in combination to control the software. Run these programs to see how the software responds to your fingers.

Mouse events

The mouse event functions are mousePressed(), mouseReleased(), mouseMoved(), and mouseDragged():

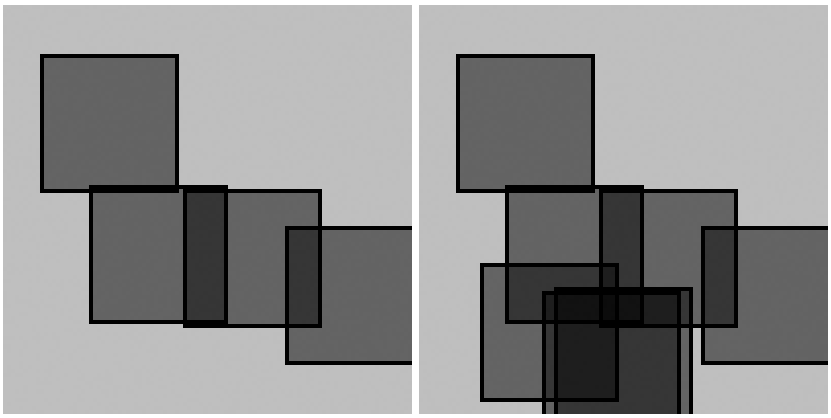
> - mousePressed() Code inside this block is run one time when a mouse button is pressed
> - mouseReleased() Code inside this block is run one time when a mouse button is released
> - mouseMoved() Code inside this block is run one time when the mouse is moved
> - mouseDragged() Code inside this block is run one time when the mouse is moved while a mouse button is pressed

Example follows:

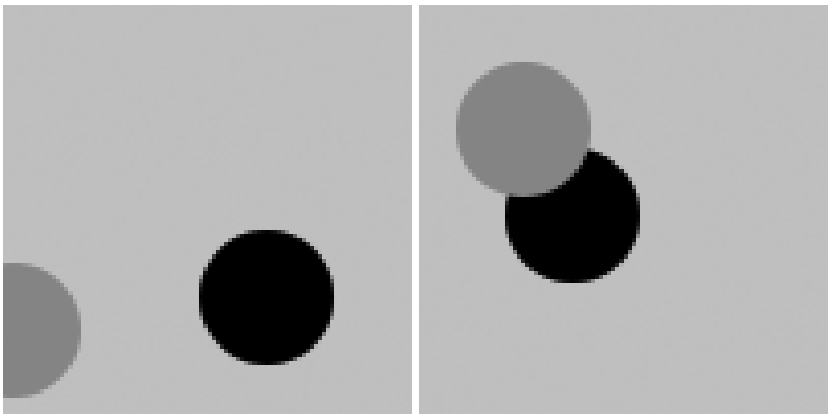
```
int gray = 0;

void setup() {
  size(100, 100);
}

void draw() {
  background(gray);
}

void mousePressed() {
  gray += 20;
}
void mouseReleased() {
  gray += 20;
}
```

show: 

```
void setup() {
  size(100, 100);
  fill(0, 102);
}

void draw() {
} // Empty draw() keeps the program running

void mousePressed() {
  rect(mouseX, mouseY, 33, 33);//creat a rectangle
}
```

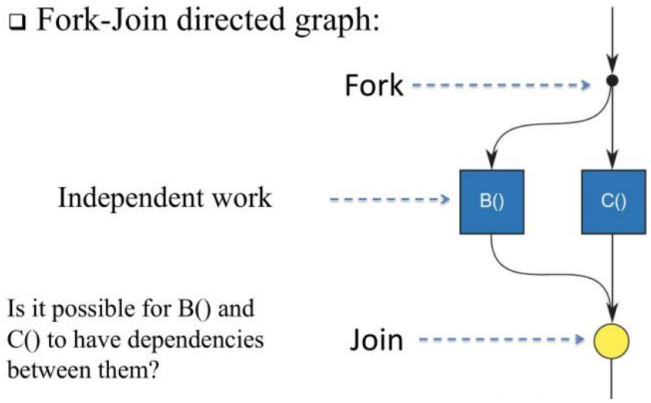


```
void mouseMoved() { // Move gray circle
  moveX = mouseX;
  moveY = mouseY;
}

void mouseDragged() { // Move black circle
  dragX = mouseX;
  dragY = mouseY;
}
```

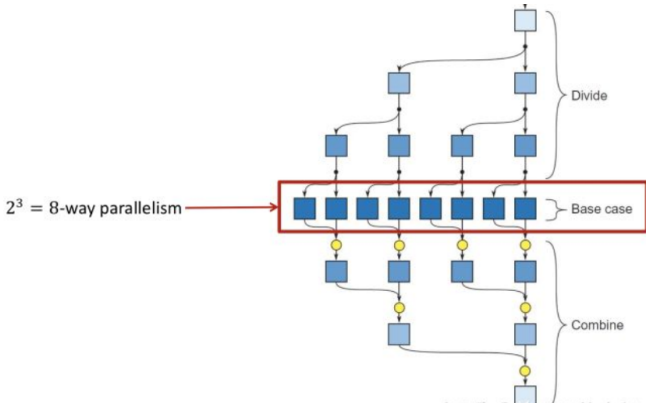


Fork/Join algorithm and Fork/Join Framework

Fork-Join algorithm



In parallel computing, the fork–join model is a way of setting up and executing parallel programs, such that execution branches off in parallel at designated points in the program, to "join" (merge) at a subsequent point and resume sequential execution. Parallel sections may fork recursively until a certain task granularity is reached.

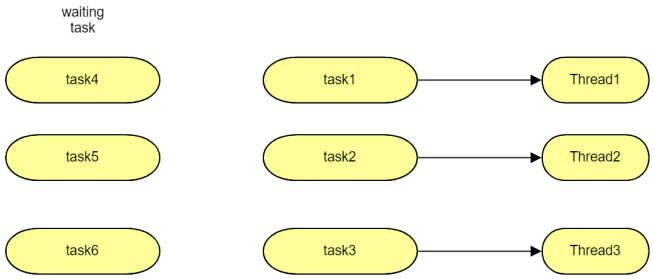


Identifying a suitable Base Case is a difficult problem

> - Forked child threads, independent of each other does not mean no dependencies
> - Recursion depth must be deep enough to have sufficient concurrency
> - But too deep (high concurrency) will cause inter-thread scheduling to become a burden in subtask processing.
> - If there are K-level N-level Fork-Joins, there will be K ^ N concurrent threads

Fork/Join Framework in Java

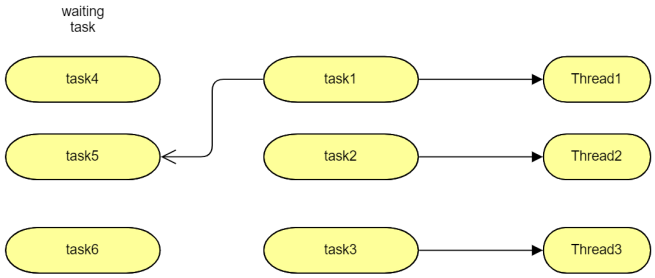
The fork/join framework was presented in Java 7. It provides tools to help speed up parallel processing by attempting to use all available processor cores – which is accomplished **through a divide and conquer approach**



Traditional thread pool

Most thread pools work this way. A thread is assigned a task, and the next task will run only if the task at hand (this task may run for a long time or may block) is completed.

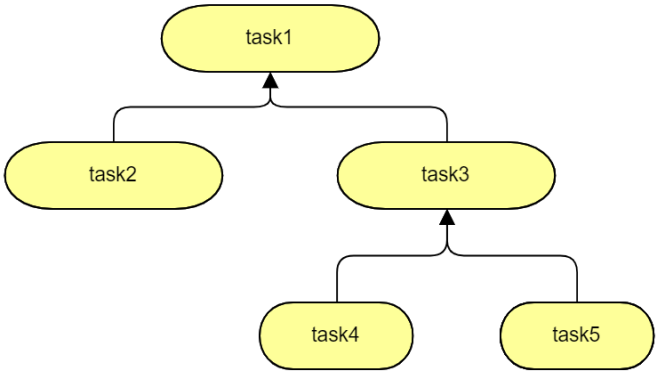
A weakness of the traditional thread pool: "When the traditional thread pool is processing the current task, and the current task depends on other tasks (subtasks)

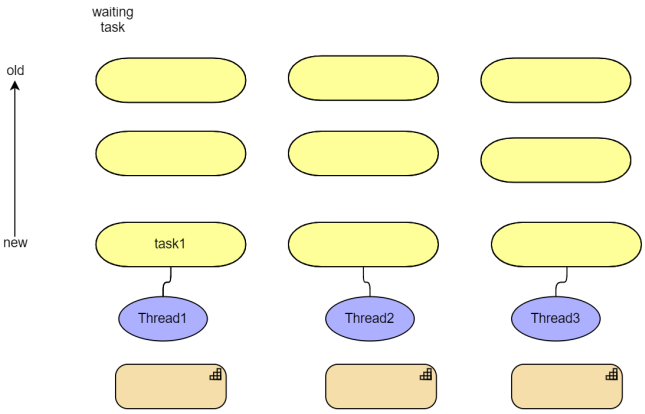


The Fork-Join framework solves the above limitation by introducing a "middle layer" between the execution thread and the task. It allows a thread to put blocked tasks aside and wait for all the sub-tasks it depends on to be processed. To handle this task blocked by subtasks. That is, suppose that "task 1" depends on "task 5" (where task 5 is a subtask created by task 1), then task 1 is set aside and performs "task 5".

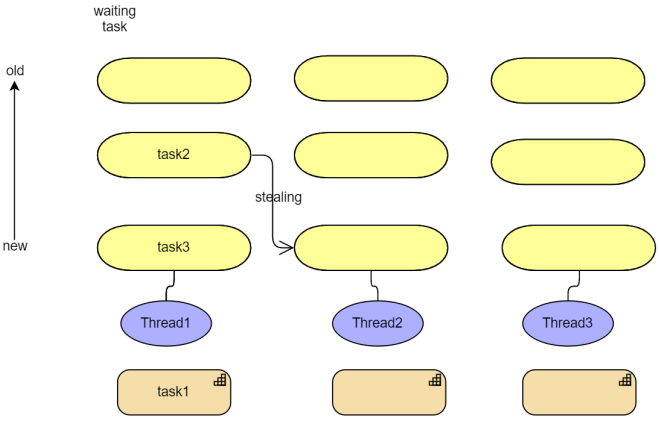
The Fork-Join framework uses ForkJoinPool, a special thread pool, to handle the above situation. It implements the "work-stealing" algorithm (workload stealing algorithm) and executes ForkJoinTask objects.

For example, calculate the task1:



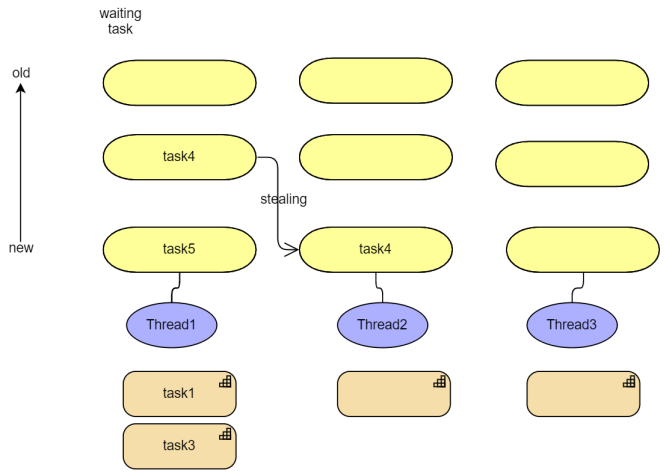


Because task1 contains task2 and task3, fork has 2 subtasks, and Thread2 "stealing" task2 , Thread1 calculates task3 as shown below:

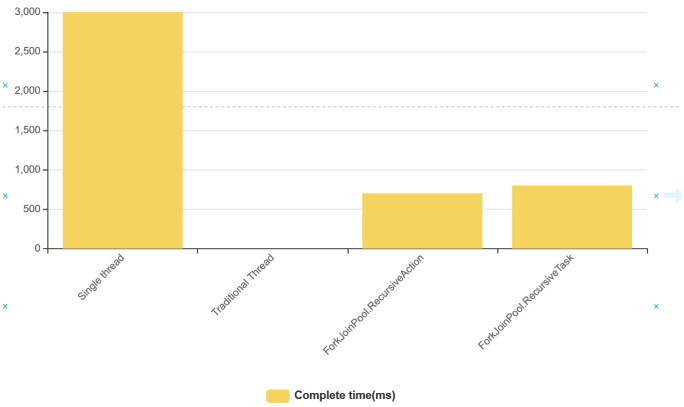


And task 1 is waiting for sub-tasks 2 and 3, so task 1 is temporarily put aside thereby releasing thread 1, the thread 1 goes to execute task 3, and thread 2 steals task 2 because it is free.

When thread 1 executes task 3, it finds that task 2 is two subtasks, task 4 and task 5, respectively, so it forks out task 4 and task 5 (the specific code will be posted below), so task 3 is put aside again.



At last ,successfully finished .(the specific code in the homework)



ForkJoinPool provides 2 types of tasks, namely "RecursiveTask" and "RecursiveAction". The RecursiveTask is a task that requires a return value and the RecursiveAction is a task that does not require a return value.

The ForkJoinPool is the heart of the framework. It is an implementation of the **ExecutorService** that manages worker threads and provides us with tools to get information about the thread pool state and performance.

Worker threads can execute only one task at the time, but the ForkJoinPool doesn't create a separate thread for every single subtask. Instead, each thread in the pool has its own double-ended queue (or deque, pronounced deck) which stores tasks.

This architecture is vital for balancing the thread's workload with the help of the work-stealing algorithm.

3. project

There is a project for Real world earthquake data visualization using Unfolding Maps library and Processing GUI.

Introduction two library of Processing and UnfoldingMaps.

In the first two weeks , the main task is that we start to get familiar to the libraries.

Base class for all sketches that use processing.core. Processing uses active mode rendering. All animation tasks happen on the "Processing Animation Thread". The setup() and draw() methods are handled by that thread, and events (like mouse movement and key presses, which are fired by the event dispatch thread or EDT) are queued to be safely handled at the end of draw().

Unfolding is a library to create interactive maps and geovisualizations in Processing and Java.

Unfolding enables you to quickly create interactive maps. Basic interactions such as Zoom & Pan are included. Simply create geo-positioned markers to display data on a map. The visual style can be adapted freely.

> - Interaction Events

> - Data Visualization

> - Styled Maps

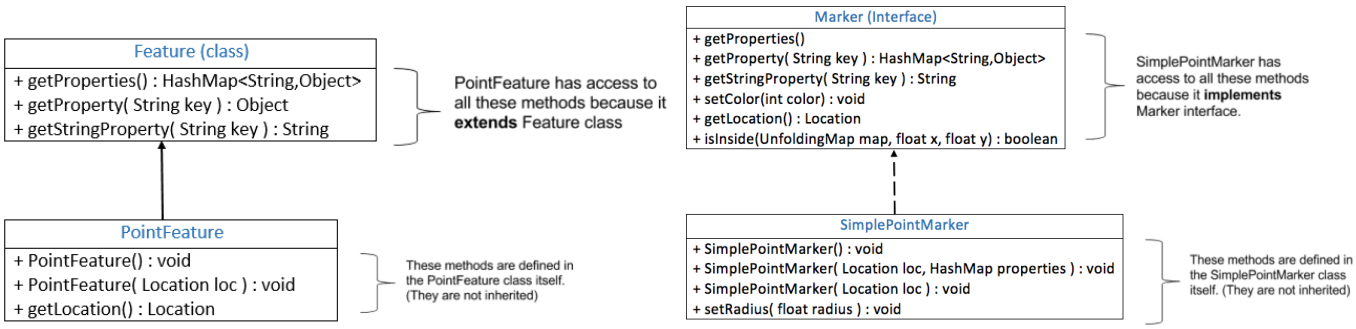
Data visualization is a powerful technique for spotting trends. In this programming assignment, i wrote a program that helps visualize earthquake data. Earthquakes are happening daily around the world.

The map reads earthquake data from a live RSS feed. (an online source of frequently updated data, see Wikipedia entry for details): http://earthquake.usgs.gov/earthquakes/feed/v1.0/summary/2.5_week.atom.

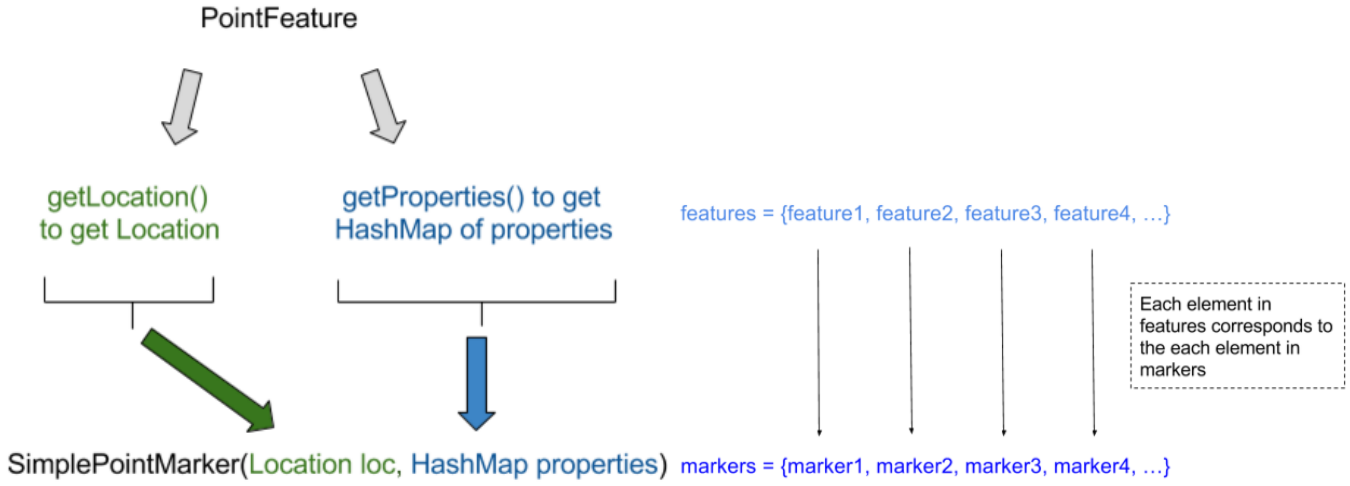
I will then plot markers in the locations where earthquakes of magnitude 2.5 or higher have occurred in the last week. Then customize the map and markers.

I organize my data into features and markers. A feature is a class of objects which stores one or more locations, its type, and any additional data properties associated with it. In this case, I will need a List of features which store the location of each earthquake in the past week, and additional properties associated with it (magnitude, age, elevation, title).

The programming assignments will rely heavily on Features and Markers. There are some relationships and dependencies in the classes.



How do we associate PointFeature and SimplePointMarker?



To display the features, i can use markers. A marker is a class of objects that give us a visual representation of a feature.

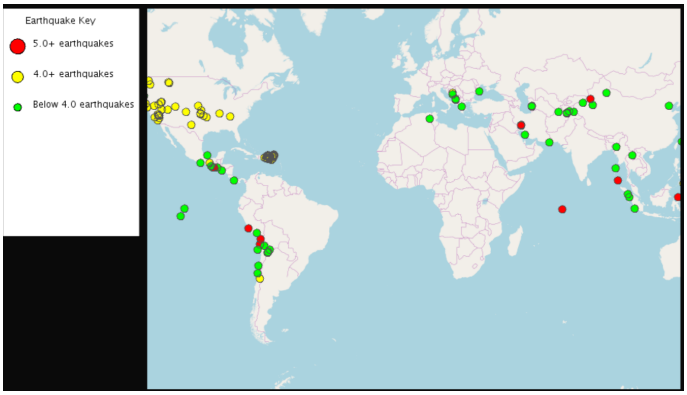
According to the magnitude of its earthquake

Minor earthquakes (less than magnitude 4.0) will have blue markers and be small.

Light earthquakes (between 4.0-4.9) will have yellow markers and be medium size.

Moderate and higher earthquakes (5.0 and over) will have red markers and be largest.

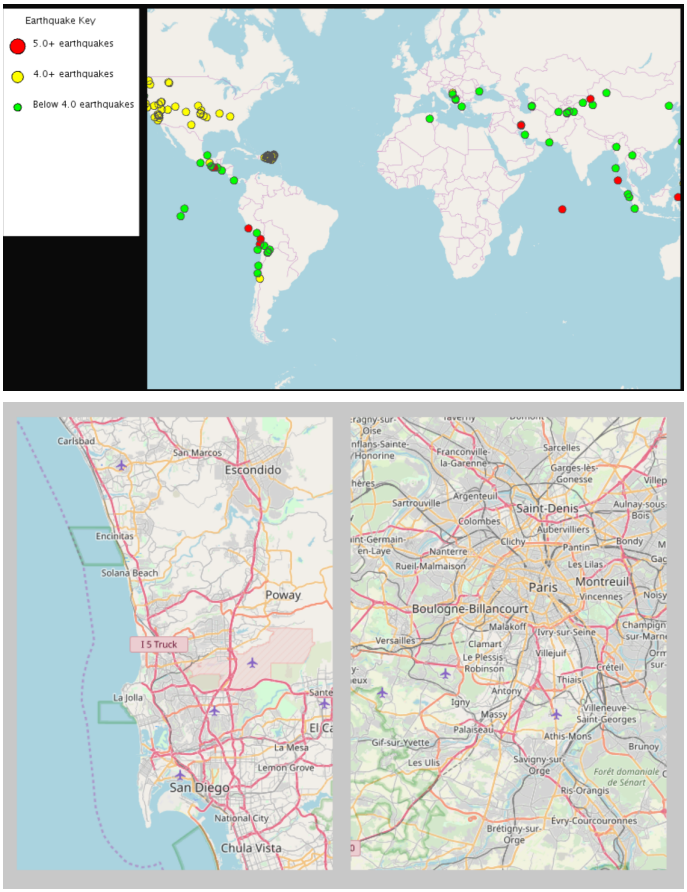
Building the interactive world map.



Here, we need use two main library, UnfoldingMaps and Processing libraries,

A small exercise for preparing our canvas. First step, This map should be zoomed into your hometown by longitude and latitude. Make a canvas on working online, feel free to try to use a different Map Provider for

each of the two maps. as we know that the GoogleMapProvider works (as do the other Google providers). The Microsoft providers (HybridProvider, AerialProvider, and RoadProvider) also are known to work.

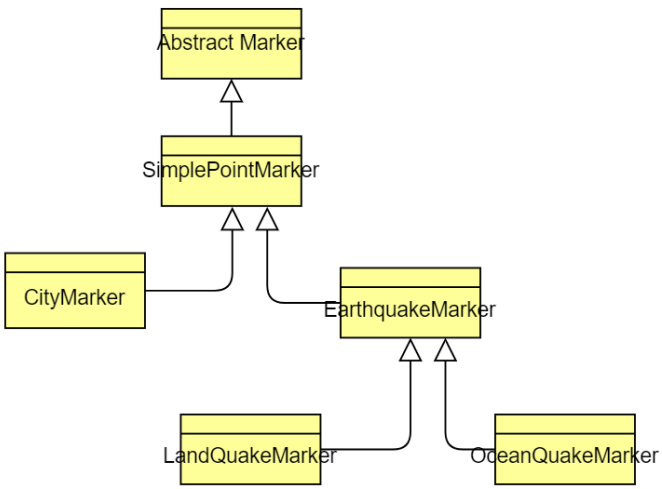


Totally,the impact of an earthquake depends on many factors, including whether the epicenter is over land or in the ocean.

The class hierarchy will allow us to customize the earthquake marker for these different kinds of earthquakes.

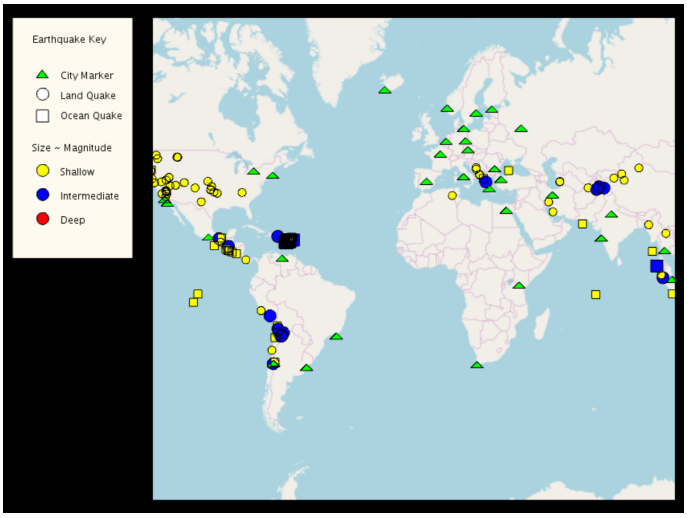
I woite code in four different new classes: an abstract class named EarthquakeMarker and two classes extending it:LandQuakeMarker and OceanQuakeMarker, as well as another class CityMarker.

In order to know the relationship,Draw a UML (class hierarchy) diagram describing the inheritance relationship between the following classes/interfaces that we need.

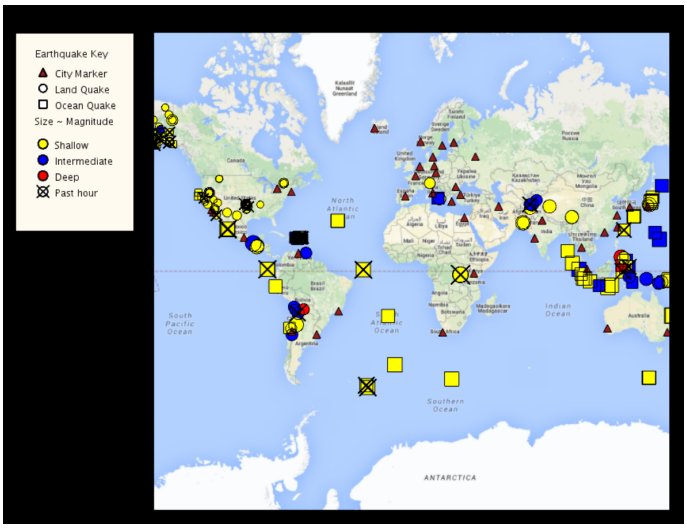


we will classifier the kinds of earthquakes by the picture. Then for distinguish the different types of earthquake in the map, we need add different colors for each other.

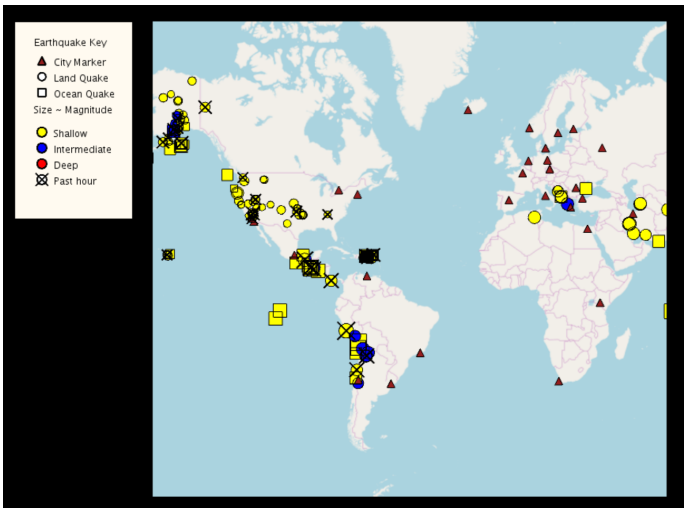
The colorDetermine() method changes the color of the earthquake marker depending on whether the earthquake is shallow, intermediate,or deep,that the size of the marker varies by its magnitude (larger magnitude=larger marker). In addition, make the LandQuakeMarkers one shape (circles) and the OceanQuakeMarkers a different shape (squares).



Modify the draw method so that it draws an X over all earthquake markers whose earthquakes have occurred in the last day



Start to interative with mouse moving and clicked.



We are going to add functionality to my map so that additional information is displayed when the user hovers over or clicks on any marker with her/ his mouse.

When she hovers over an earthquake marker, my map will display the title of the earthquake (including its magnitude and region). Clicking on a marker gives even more information: A click on a city marker will lead to only that city and earthquakes which affect it being displayed on the map. Clicking once again on that marker will bring the rest of the map's markers back. Similarly, after clicking on an earthquake marker, only cities potentially affected by that earthquake will be displayed.

I will use event-driven programming to make this happen.

Processing 

Use sorting to answer questions about data sets

In the final piece of the project, you will organize the earthquake data and compute statistics on it. ,it sorts earthquakes in reverse order of magnitude. Implement the Comparable interface SortandPrint

4.Conclusion

This Specialization covers intermediate topics in software development. i begin to learn object-oriented programming principles that will allow to use Java to its full potential, and implement data structures and algorithms for organizing large amounts of data in a way that is both efficient and easy to work with.

By the end of this course and I feels empowered to create a Java program that’s more advanced than any i have created in the past and that is personally interesting. I also practice critically evaluating my own code.

In achieving this goal I have learned the fundamentals of Object Oriented Programming, how to leverage the power of existing libraries and how to make full use of java's documents, how to build graphical user interfaces, and how to use some core algorithms for searching and sorting data. And this course is project-based and fundamental, so i am so happy to choose it!

I also learned a basic knowledge of sequential programming in Java and learn foundational topics in Parallelism, who is motivated to learn how to write parallel, concurrent and distributed programs.