개념 및 TIP

```java
public class _99th_Practice {
    public static void main(String[] args) {
        for (char ch = 'A'; ch < 'Z'; ch++) {
            System.out.print(ch);
            if (ch == 'O') {
                System.out.println();
            }
        }
        System.out.println();
        for (char k = 'ㄱ'; k < 'ㅎ'; k++) {
            System.out.print(k);
            if (k == 'ㄹㄹ') {
                System.out.println();
            }
        }
    }
}
```

```
"C:\Program Files\Java\jdk-1
ABCDEFGHIJKLMNO
PQRSTUVWXY
ㄱㄲㄳㄴㄵㄶㄷㄸㄹㄺㄻㄼㄽㄾ
ㄿㅀㅁㅂㅃㅄㅅㅆㅇㅈㅉㅊㅋㅌㅍ
Process finished with exit c
```

〈char로 for문 가능〉

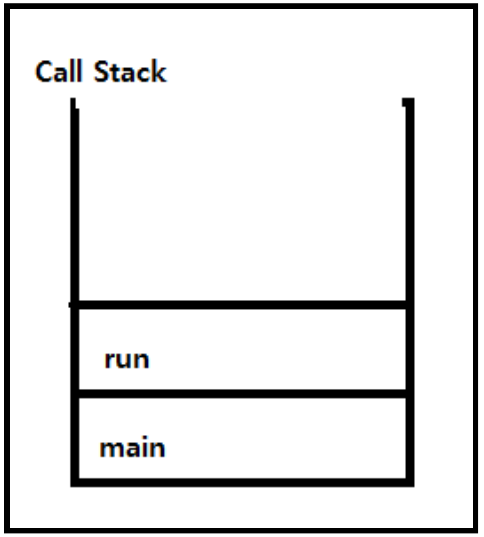
ch++ 》 A 〉 B 〉 C 〉 D 〉 .....

k++ 》 ㄱ〉...〉...〉ㅎ

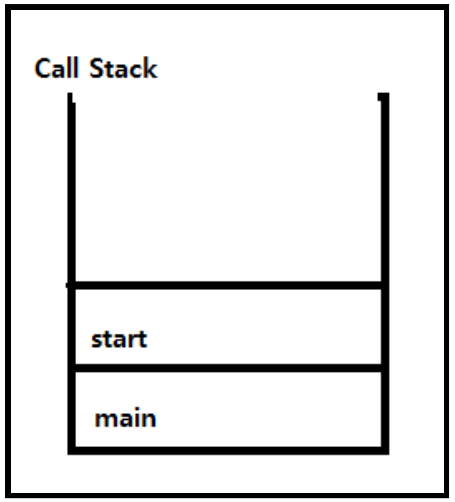
〈Thread 사용시 run()과 start()〉

쓰레드를 실행시킬 때, run()이 아닌 start()를 호출.
 ›› main 메서드에서 run()을 호출하는 것은 생성된 쓰레드를 실행시키는 것이 아니라 단순히
클래스에 선언된 메서드를 호출하는 것일 뿐.

반면 start()는, 새로운 쓰레드가 작업을 실행하는데 필요한 호출스택(call stack)을 생성한 다음에
run()을 호출해서, 생성된 호출스택에 run()이 첫 번째로 올라가게 된다.
모든 쓰레드는 독립적인 작업을 수행하기 위해 자신만의 호출스택을 필요로 하기 때문에
새로운 쓰레드를 생성하고 실행시킬 때마다 새로운 호출스택이 생성되고 쓰레드가 종료되면 작업에 사용된 호출스택은 소멸됩니다.

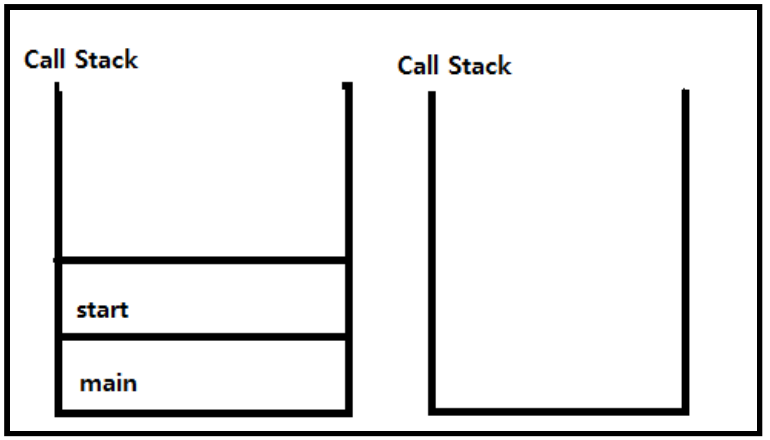


main method에서 run()을 호출했을 때의 call stack.



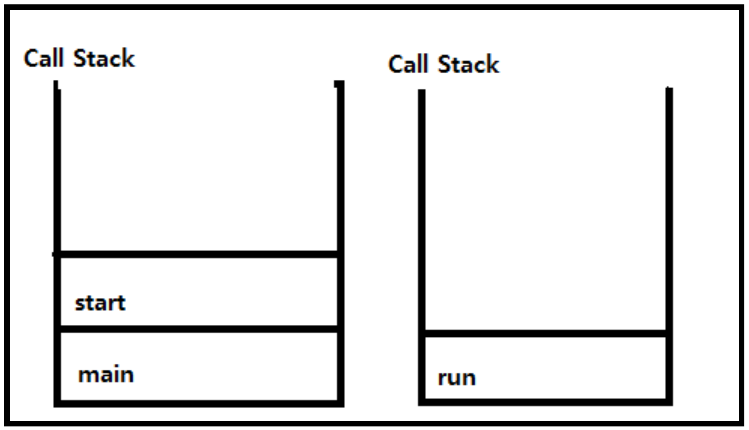
1) main method에서 start()를

호출했을 때의 call stack



2) start()는 새로운 쓰레드를 생성하고.

쓰레드가 작업하는데 사용될 호출스택을 생성.

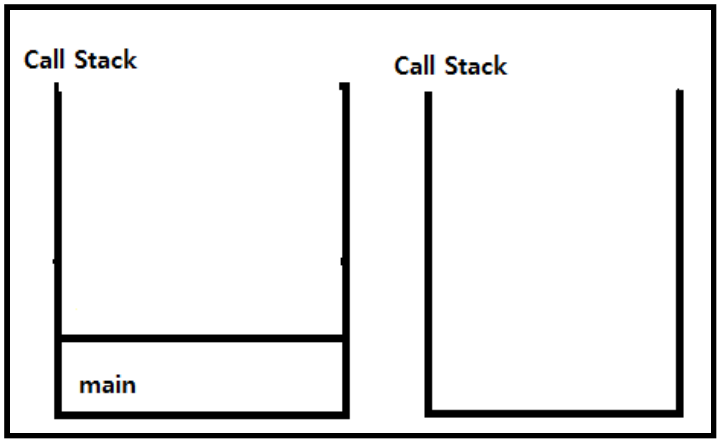


3) 새로 생성된 호출스택에 run()이 호출되어

쓰레드가 독립된 공간에서 작업을 수행.

4) 이제는 호출스택이 2개이므로

스케줄러가 정한 순서에 의해서 번갈아 가면서 실행.



호출스택에서는 가장 위에 있는 메써드가 현재 실행중인 메써드이고 나머지 메써드들은 대기상태에 있는다.

그러나 위의 그림에서와 같이 쓰레드가 둘 이상일 때는 호출스택의 최상위에 있는 메써드일지라도 대기상태에 있을 수 있다.

스케줄러는 실행 대기중인 쓰레드들의 우선순의를 고려하여 실행순서와 실행시간을 결정하고.

각 쓰레드들은 작성된 스케줄에 따라 자신의 순서가 되면 지정된 시간동안 작업을 수행.

이 때 주어진 시간동안 작업을 마치지 못한 쓰레드는 다시 자신의 차례가 돌아올 때까지 대기상태로 있게 되며.

작업을 마친 쓰레드. 즉 run()의 수행이 종료된 쓰레드는 호출스택이 모두 비워지면서 이 쓰레드가 사용하던 호출스택은 사라짐.

이는 마치 자바프로그램을 실행하면 호출스택이 생성되고 main메서드가 처음으로 호출되고.

main메서드가 종료되면 호출스택이 비워지면서 프로그램도 종료되는 것과 같다.

출처: 스레드 예제포함

출처: https://blog.naver.com/opgj123/221408913970

- 지속적으로 UPDATE 할 것 -