

JAVA 로 컴파일된 코드의 기능들 R에서 사용하기

윈도우 환경에서 Java로 컴파일(Compile)된 코드를 R에서 사용하기 위해서는 rJAVA를 이용한 Java Code 호출한다.

시작숫자와 끝숫자를 매개변수로 하여 두 번호 간 누적합을 구하는 Java 예제를 작성하고, 해당 예제를 compile 한 파일을 R에서 호출하여 1부터 10까지의 누적합을 산출해 본다.

<- 시스템 환경 >

Window 7, 64bit
R 3.0.1, 64bit
Java SE 1.7.0_17, 64bit

1. Java Code 작성 및 Compile

실행할 Java 코드를 작성하여, “D:\demo\StartEndSum.java”에 저장한다.

코드에서 시작숫자와 끝숫자간의 누적합을 구하는 핵심 기능은 getSum 메서드로 분리하였으며, main 메서드에서는 getSum 메서드를 호출하여 인자로 받은 두 숫자간 누적합을 출력하는 보조적 기능을 수행한다.

여기에서 main 메서드는 컴파일 후 code가 제대로 기능하는지 검증하는 역할만 담당하며, 실제로 R에서는 getSum 메서드를 직접 호출하여 사용함을 유의한다.

```

class startEndSum
{
    public static void    main(String[] args)
    {
        if (args.length != 2) {
            System.err.println("Usage: java startEndSum <startNum> <endNum>");
            System.exit(1);
        }
        int startNum=Integer.parseInt(args[0]);
        int endNum=Integer.parseInt(args[1]);

        startEndSum    startEndSum = new startEndSum();

        int total=    startEndSum.getSum(startNum, endNum);
        System.out.println("total : " + total);
    }

    public int getSum(int    start, int end) {

        int totalCount = 0;
        for (int i = start; i <= end; i++)
        {
            totalCount += i;
        }
        return totalCount;
    }
}

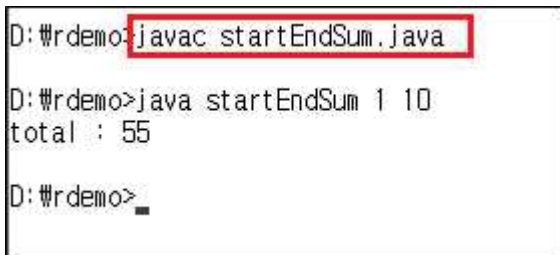
```

123.java

윈도우의 cmd창에서 Java Code 파일 디렉토리에 접근 후 아래 명령을 실행하여 자바 파일을 컴파일한다.
컴파일이 완료되면 정상 실행되는지 테스트한다. 매개변수로 1과 10을 입력하여, 1부터 10까지의 누적합 55가 출력되는지 확인해 본다 .

```
# 윈도우 커맨드창에서 실행
$ javac startEndSum.java

$ java startEndSum 1 10
total : 55
```



```
D:\Wdemo>javac startEndSum.java

D:\Wdemo>java startEndSum 1 10
total : 55

D:\Wdemo>_
```

2. R 사전 설정

이제 Java 는 준비가 완료되었으므로, R을 실행한 다음 rJava를 설치한후 로드 한다. jvm을 초기화 한다.
다음은 컴파일한 Java class 의 파일 경로를 classpath에 추가한다. classpath 목록에 컴파일했던 경로 “D:\Wdemo” 디렉토리가 있는 지 확인한다. 정상적으로 초기화될 경우 아무 문구도 출력되지 않거나, “ [1] 0 “ 이라는 메시지가 출력된다. Java Code 작성 및 Compile’에서 생성한 java class파일의경로를 classpath에 추가한다(디렉토리 구분이 역슬러시 2개임에 유의한다).

```
# R 에서 실행
> install.packages("rJava")
> library(rJava)

> jnint() # JVM을 초기화
[1] 0

> jaddClassPath("D:\Wdemo")
```

아래의 명령을 실행하여 classpath가 정상적으로 추가되었는지 확인한다.

```
> print(jclassPath()) # classpath가 정상적으로 추가되었는지 확인
[1] "D:\Wdemo" # 작성한 자바 파일이 들어있는 디렉토리 확인
```

```

> library(rJava)
> .jinit()
[1] 0
> .jaddClassPath("D:\\rdemo")
> print(.jclassPath())
[1] "C:\\Users\\LeeKiYoung\\Documents\\R\\win-library\\3.0\\rJava\\java"
[2] "."
[3] "C:\\Program Files\\Java\\jdk1.7.0_17\\lib"
[4] "C:\\Program Files\\Java\\jdk1.7.0_17\\lib\\tools.jar"
[5] "D:\\rdemo"
> |

```

3. Java Class파일 호출

컴파일한 java class 파일의 객체를 생성한 후 임의의 변수에 저장한다.

```
> obj <- jnew("startEndSum")
```

아래 명령을 실행하여, getSum 메서드를 호출하고 인자를 아래와 같이 준다

getSum 메서드를 호출하고 입력변수를 아래와 같이 지정한다.

위 명령에서 jcall 함수의 첫번째 인자는 java class파일 객체를 저장한 변수이며,

두 번째 인자 "1"은 호출하는 java method의 반환형이 정수형(Integer)임을 나타내는 약어이다.

세 번째 인자는 java class 파일에서 호출하는 메서드명을,

네 번째와 다섯 번째의 as.integer(1),as.integer(10)은 getSum 메서드에서 필요로 하는 인자값들을 나타낸다.

R에서 일반 숫자는 number형이므로, java의 integer 자료형과 일치시키려면 as.integer() 로 변환해야 한다.

호출 및 연산이 정상 수행되었을 경우 아무 메시지도 출력되지 않는다.

결과값은 result 변수에 저장되었으므로, 해당 변수를 명령어처럼 실행하면 결과가 출력된다.

```

> result <- jcall(obj,"1","getSum",as.integer(1),as.integer(10))

> result
[1] 55

```

```
> obj <- .jnew("startEndSum")
> result <- .jcall(obj,"I","getSum",as.integer(1),as.integer(10))
> result
[1] 55
>
```

4. rJava 패키지가 설치되지 않을 경우

Java 가 연동되지 않으면 가장 먼저 할 일은 Java를 재설치 해보고, R을 다시 설치한 후 컴퓨터를 재부팅하는 것이다. Java를 새로운 버전으로 다운받아 설치하려면 다음을 참조한다.

<http://www.oracle.com/technetwork/java/javase/downloads/jdk7-downloads-1880260.html>

rJava 패키지가 설치되지 않을 경우 윈도우 환경변수의 path를 다음과 같이 수정한다. path를 수정하는 방법은 다음과 같다. Java 의 jre를 설치한 경우는 다음과 같다.

```
path <- C:\Program Files\Java\jre6\bin\ (remove "client").
path <- C:\Program Files\RR-2.10.1\bin\
remove path C:\Program Files\RR
```

자바에 접근 가능한 지 윈도우의 커맨드창을 띄워 확인한다.

```
$ java -version
```

윈도우의 커맨드창을 띄우기가 귀찮은 경우 R에서 간단하게 확인하는 방법도 있다.

```
> system("java -version").
```

jre 가 아닌 jdk 를 설치하였다면 시스템 환경 path 를 지정하는 곳에서 “JAVA_HOME” 이라는 변수를 설정한 후 jdk 가 포함된 디렉토리를 지정한다. jre 는 컴파일된 자바 프로그램을 실행하는 자바환경 디렉토리이다. jdk 는 Java 개발도구이다. 자바프로그램을 할 때 필요한 컴파일러 등이 들어있다. 자바로 프로그램을 하지 않을 사용자라면 다운로드 받을 필요가 없다. jdk 를 설치하면 jre 는 자동으로 설치된다.

이와 같이 실행한 후에도 계속하여 오류가 난다면 Java 와 R이 동일하게 32비트 혹은 64비트인지 확인한다. 만약 jvm.dll 파일을 찾을 수 없다는 오류가 난다면, 환경변수 path에 jvm.dll 이 포함된 디렉토리명(예, “C:\Users\computer_name\Documents\R\win-library\3.0\rJava\jri\x64;”)을 추가해 준다.