



# 네트워킹



# 네트워킹이란

2



## ○ 네트워킹

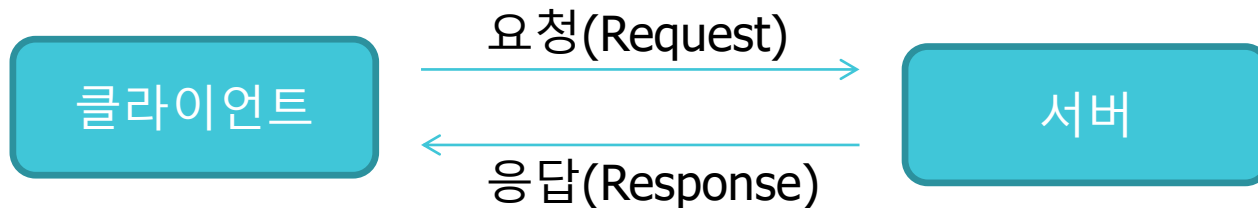
- ① 인터넷 망에 연결되어 있는 원격지의 서버 또는 원격지의 단말과 통신을 통해 데이터를 주고 받는 일반적인 일
- ② 최근 정보를 주고 받기 위해 많이 사용하는 소셜 네트워크서비스 (SNS)가 네트워킹을 활용하는 대표적인 예



## ○ 원격지의 서버를 연결하는 방식

### ◎ 2-tier C/S 모델

- 클라이언트와 서버가 일대일로 연결하는 방식



### ◎ 3-tier 모델

- 서버를 좀 더 유연하게 구성
- 응용 서버와 데이터 서버로 구성하는 경우, 데이터베이스를 분리시킴



# 소켓사용하기



## ○ 네트워크에 대한 이해

- TCP/IP 수준의 통신 방식을 제공하는 소켓을 이용해 서버에 연결해 보면 이해하기 쉬움
- 일반적인 프로그래밍에서는 대부분 TCP 연결 사용
- 비연결성 특성으로 인해 실시간으로 데이터를 처리하는 애플리케이션의 경우, 응답 속도를 높이기 위해 HTTP 보다 소켓 연결 선호

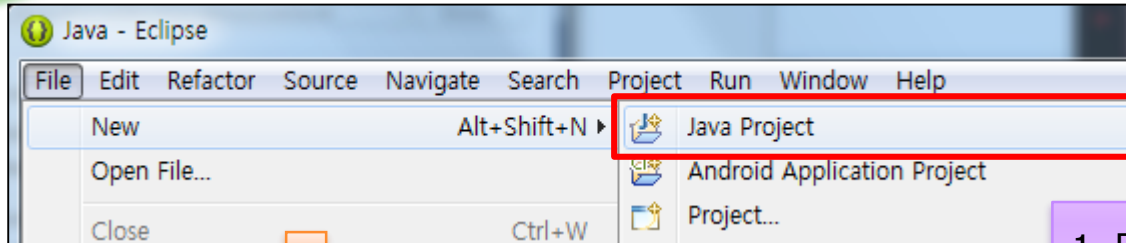
## ○ 소켓 연결 방식

- 안드로이드에서는 표준 자바의 소켓을 그대로 사용할 수 있음
- 서버 : 서버 소켓을 만들어 실행함(포트 지정)
- 클라이언트: 소켓을 만들어 서버 소켓으로 연결함(IP와 포트 지정)
- Stream 객체를 이용해 데이터를 보내거나 받을 수 있음

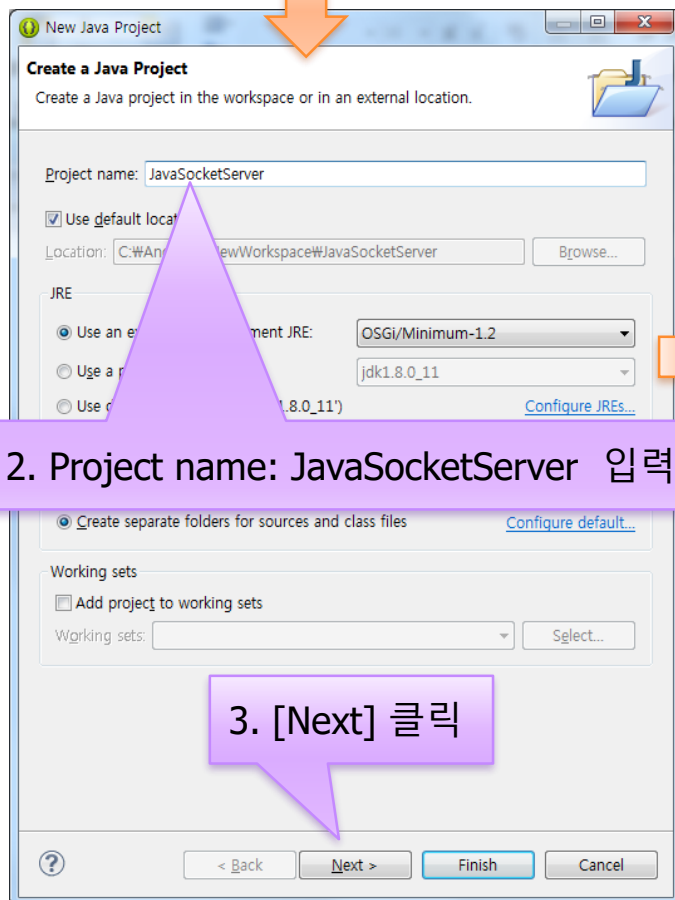


# 자바 소켓 프로그램

5

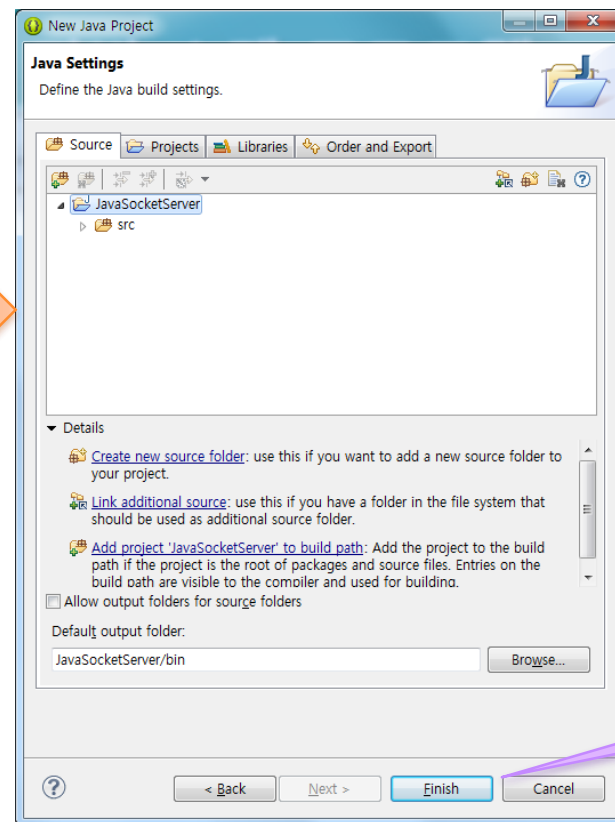


1. [File]→[New]→[Java Project] 클릭

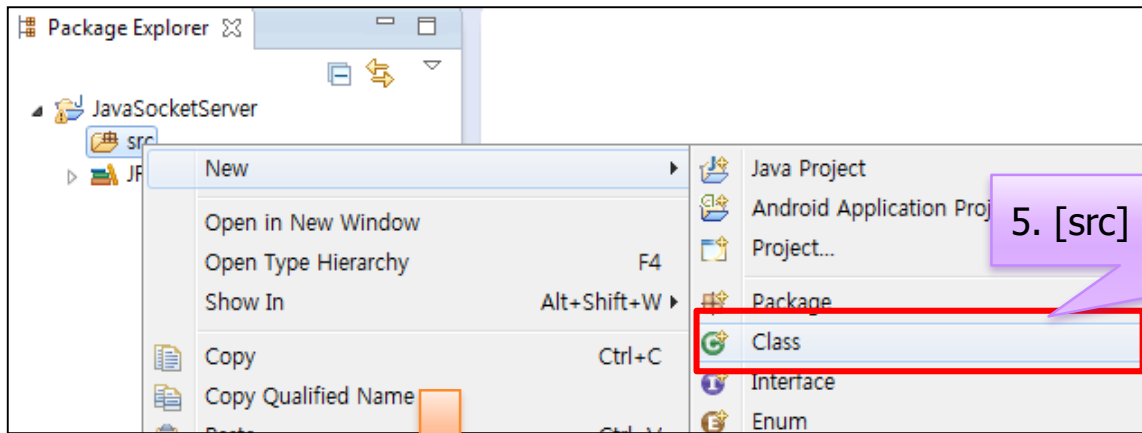


2. Project name: JavaSocketServer 입력

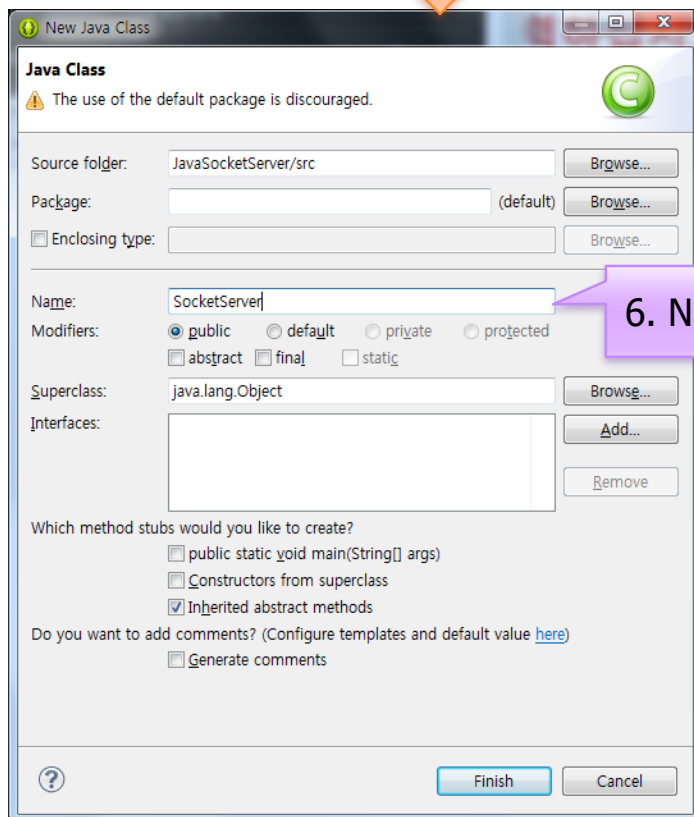
3. [Next] 클릭



4. [Finish] 클릭



5. [src] 오른쪽 마우스 클릭 후 [class] 클릭



6. Name: SocketServer 입력

```
6 import java.net.Socket;
7
8 public class SocketServer {
9
10     public static void main(String[] args) {
11         try {
12             int portNumber = 5001;
13
14             System.out.println("Starting Java Socket Server ...");
15
16             ServerSocket aServerSocket = new ServerSocket(portNumber);
17             System.out.println("Listening at port " + portNumber + "...");
18
19             while(true) {
20                 Socket sock = aServerSocket.accept();
21                 InetAddress ClientHost = sock.getLocalAddress();
22                 int clientPort = sock.getPort();
23                 System.out.println("A client connected. host : " +
24                                     ClientHost + ", port : " + clientPort);
25                 ObjectInputStream instream = new ObjectInputStream(sock.getInputStream());
26                 Object obj;
27                 try {
28                     obj = instream.readObject();
29                     System.out.println("Input : " + obj);
30
31                     ObjectOutputStream outstream = new ObjectOutputStream(sock.getOutputStream());
32                     outstream.writeObject(obj + " form Server");
33                     outstream.flush();
34                     sock.close();
35                 } catch (ClassNotFoundException e) {
36                     // TODO Auto-generated catch block
37                     e.printStackTrace();
38                 }
39             }
40         } catch (IOException ex) {
41             // TODO Auto-generated catch block
42             ex.printStackTrace();
43         }
44     }
45 }
46 }
```

7. 코딩



```
8 public class SocketServer {
9
10     public static void main(String[] args) {
11         try {
12
13             int portNumber = 5001;
14
15             System.out.println("Starting Java Socket Server ...");
16
17             ServerSocket aServerSocket = new ServerSocket(portNumber);
18             System.out.println("Listening at port " + portNumber + "...");
19
20             while(true) {
21                 Socket sock = aServerSocket.accept();
22                 InetAddress ClientHost = sock.getLocalAddress();
23                 int clientPort = sock.getPort();
24                 System.out.println("A client connected. host : " +
25                                     ClientHost + ", port : " + clientPort);
26                 ObjectInputStream instream = new ObjectInputStream(sock.getInputStream());
27                 Object obj;
28                 try {
29                     obj = instream.readObject();
30                     System.out.println("Input : " + obj);
31
32                     ObjectOutputStream outstream = new ObjectOutputStream(sock.getOutputStream());
33                     outstream.writeObject(obj + " form Server");
34                     outstream.flush();
35                     sock.close();
36                 } catch (ClassNotFoundException e) {
37                     // TODO Auto-generated catch block
38                     e.printStackTrace();
39                 }
40             }
41         } catch (IOException ex) {
42             // TODO Auto-generated catch block
43             ex.printStackTrace();
44         }
45     }
46 }
```

서버 소켓 생성

클라이언트가 접속할 때 소켓 개체 생성

반복문으로 클라이언트 연결 대기

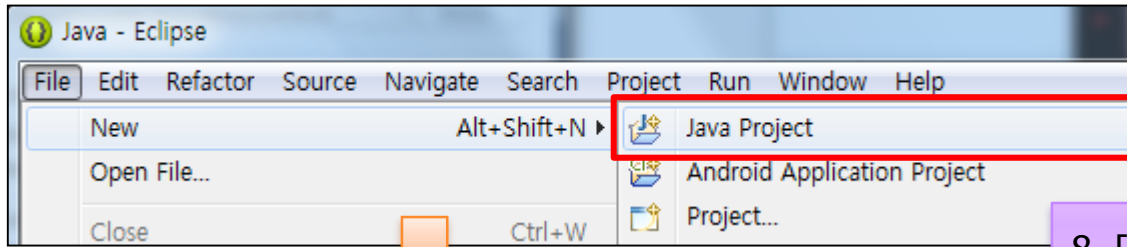
클라이언트가 보낸 오브젝트를 받는다.

클라이언트에 오브젝트를 보낸다.

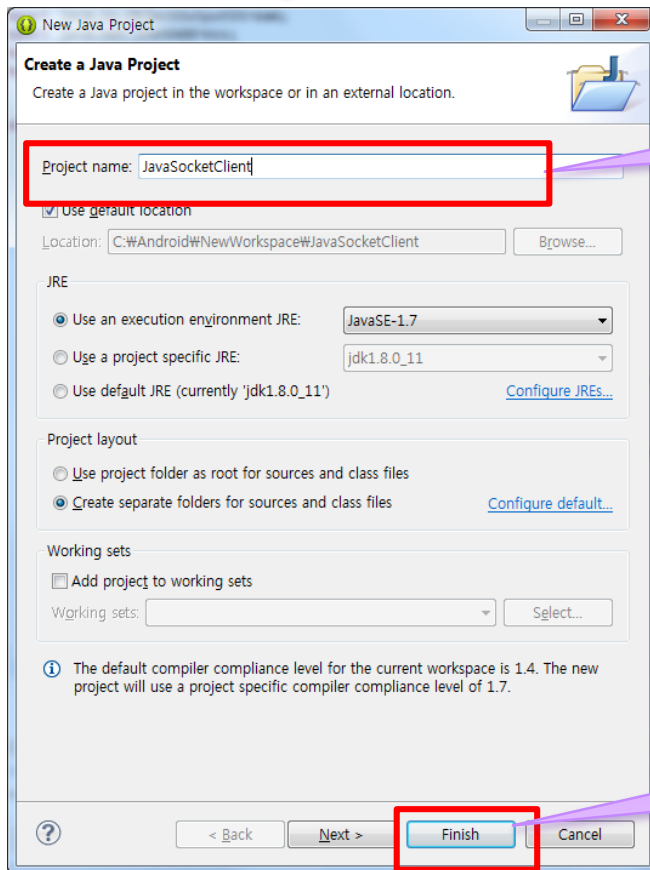
클라이언트에 오브젝트를 보낸다.



# 클라이언트 만들기

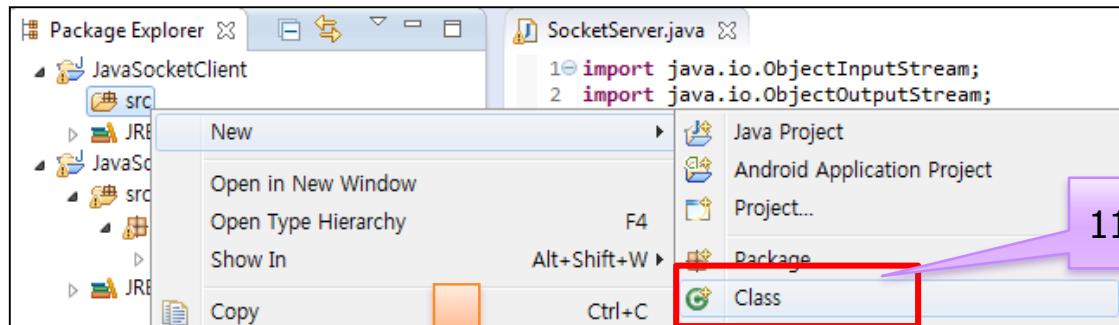


8. [File]→[New]→[Java Project] 클릭

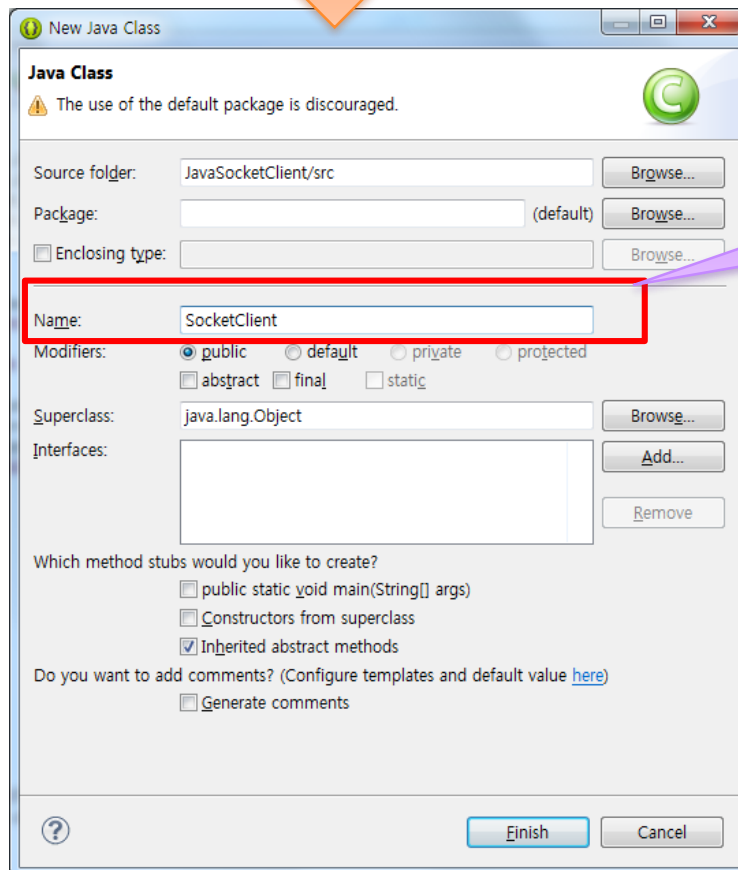


9. "Project name: JavaSocketClient" 입력

10. 클릭



11. [src] 오른쪽 마우스 클릭 후 [Class] 클릭



12. Name : SocketClient 입력

```
SocketClient.java ✕  
1 import java.net.Socket;  
2  
3  
4 public class SocketClient {  
5     public static void main(String[] args){  
6         int portNumber = 5001;  
7         Socket sock = new Socket("localhost", portNumber);  
8  
9  
10    }  
11 }
```

13. 클릭

```
SocketClient.java ✕  
1 import java.net.Socket;  
2  
3  
4 public class SocketClient {  
5     public static void main(String[] args){  
6         int portNumber = 5001;  
7         Socket sock = new Socket("localhost", portNumber);  
8  
9  
10    }  
11 }  
12
```

14. 클릭

Unhandled exception type  
3 quick fixes available:  
[Add throws declaration](#)  
[Surround with try/multi catch](#)  
[Surround with try/catch](#)

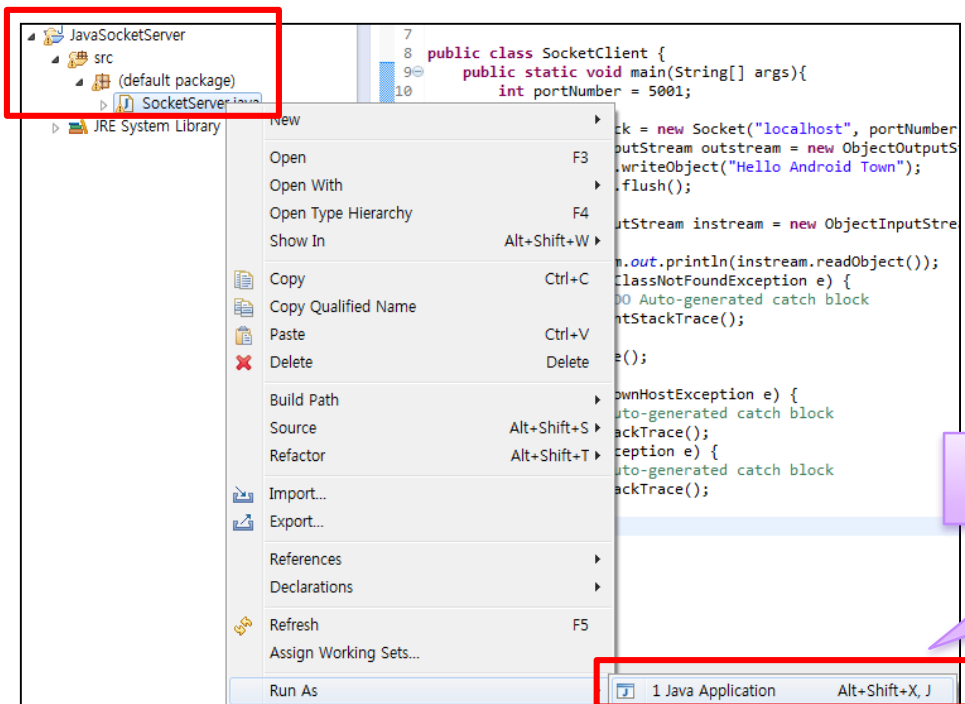
15. 코딩

```
1 import java.io.IOException;
2 import java.io.ObjectInputStream;
3 import java.io.ObjectOutputStream;
4 import java.net.Socket;
5 import java.net.UnknownHostException;
6
7
8 public class SocketClient {
9     public static void main(String[] args){
10         int portNumber = 5001;
11         try {
12             Socket sock = new Socket("localhost", portNumber);
13             ObjectOutputStream outstream = new ObjectOutputStream(sock.getOutputStream());
14             outstream.writeObject("Hello Android Town");
15             outstream.flush();
16
17             ObjectInputStream instream = new ObjectInputStream(sock.getInputStream());
18             try {
19                 System.out.println(instream.readObject());
20             } catch (ClassNotFoundException e) {
21                 // TODO Auto-generated catch block
22                 e.printStackTrace();
23             }
24             sock.close();
25
26         } catch (UnknownHostException e) {
27             // TODO Auto-generated catch block
28             e.printStackTrace();
29         } catch (IOException e) {
30             // TODO Auto-generated catch block
31             e.printStackTrace();
32         }
33     }
34 }
35 }
```

```
8 public class SocketClient {
9     public static void main(String[] args){
10         int portNumber = 5001;
11         try {
12             Socket sock = new Socket("localhost", portNumber);
13             ObjectOutputStream outstream = new ObjectOutputStream(sock.getOutputStream());
14             outstream.writeObject("Hello Android Town");
15             outstream.flush();
16
17             ObjectInputStream instream = new ObjectInputStream(sock.getInputStream());
18             try {
19                 System.out.println(instream.readObject());
20             } catch (ClassNotFoundException e) {
21                 // TODO Auto-generated catch block
22                 e.printStackTrace();
23             }
24             sock.close();
25
26         } catch (UnknownHostException e) {
27             // TODO Auto-generated catch block
28             e.printStackTrace();
29         } catch (IOException e) {
30             // TODO Auto-generated catch block
31             e.printStackTrace();
32         }
33     }
34 }
35 }
```

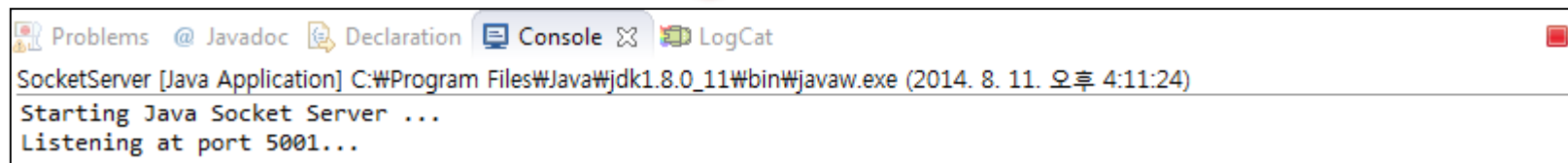
서버에 오브젝트를 보낸다.

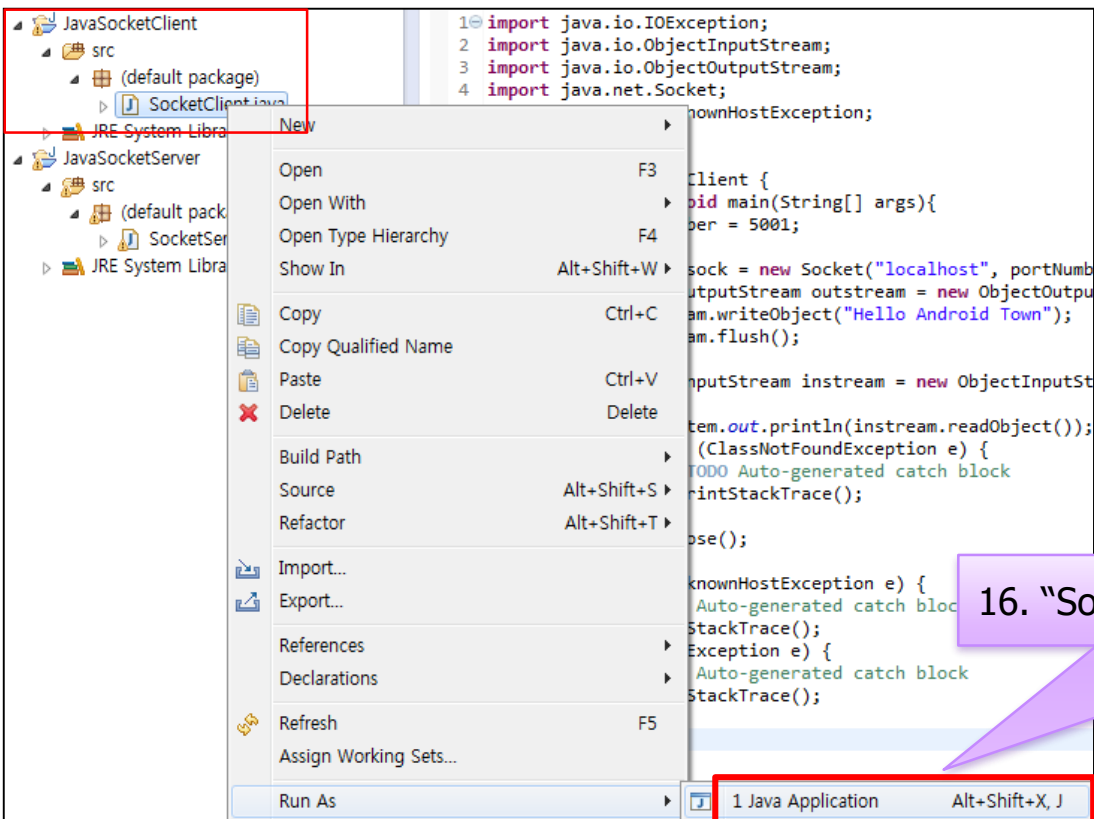
서버가 보낸 오브젝트를 받는다.



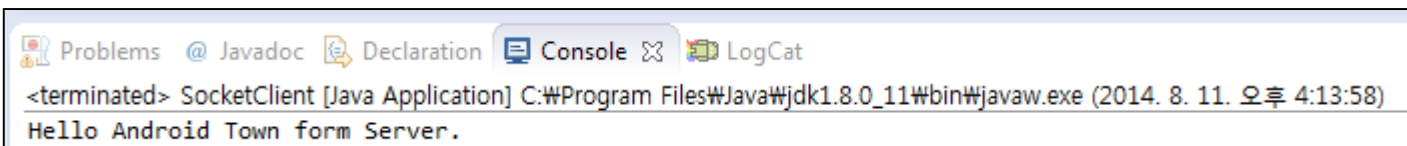
15. "SocketServer"오른쪽 마우스 클릭 후 실행

실행

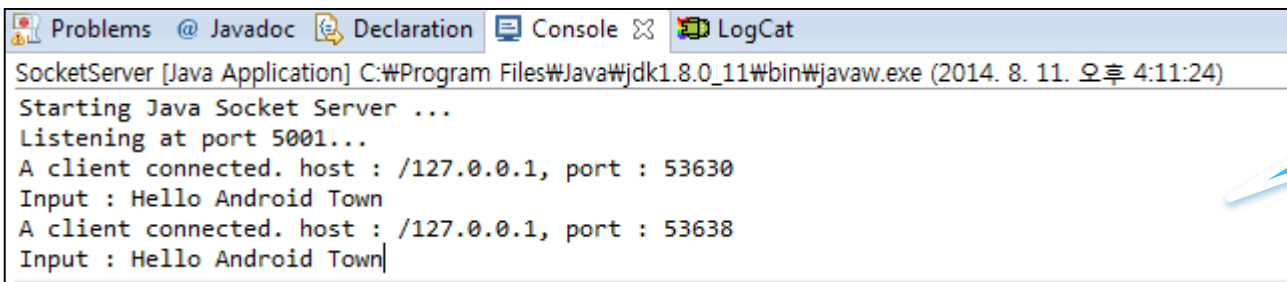




Socket Client 영역

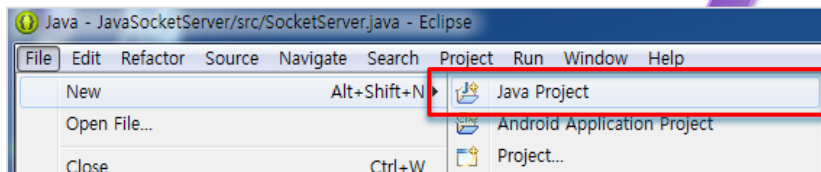


Socket Server 영역



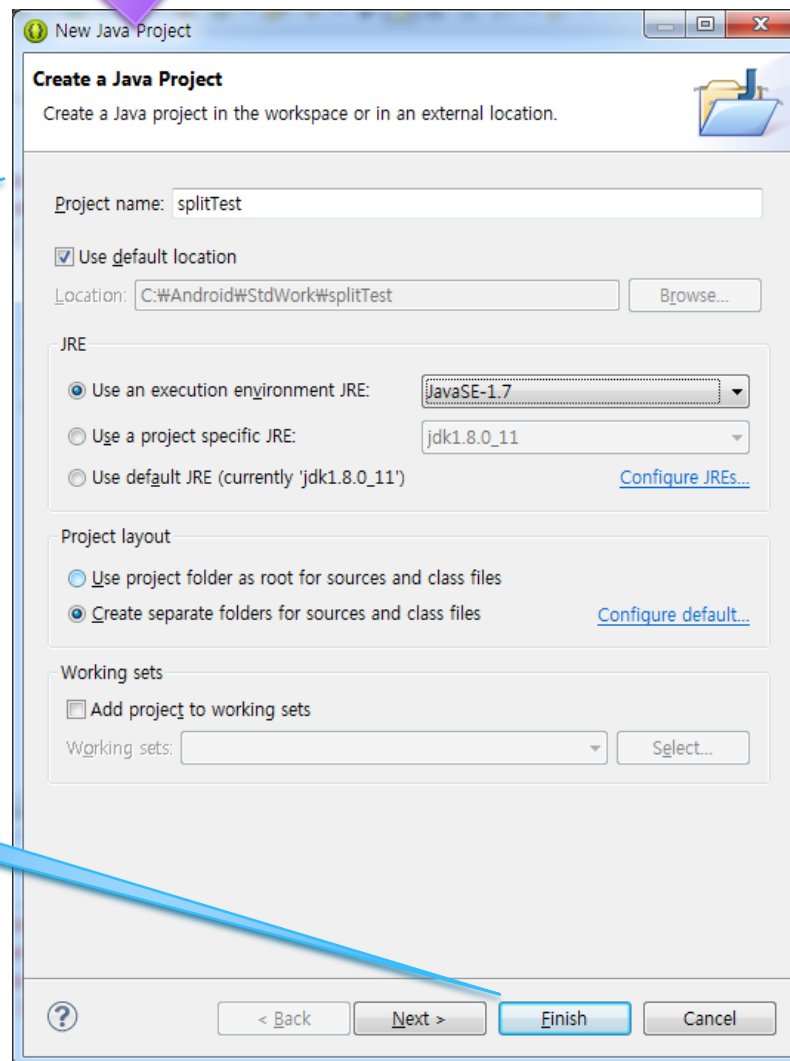


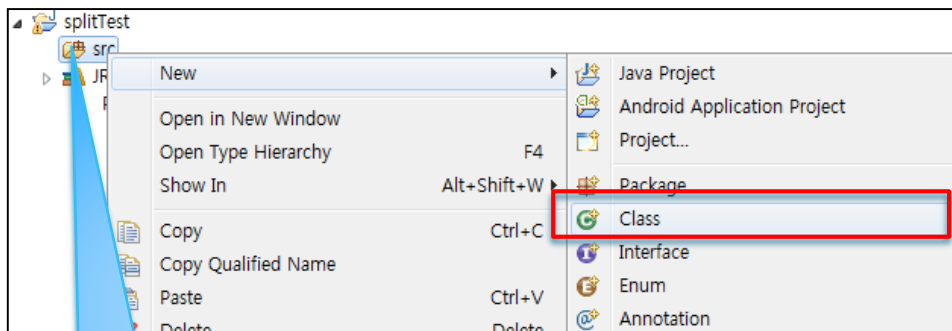
# 이클립스 자바 프로젝트 생성



자바 프로젝트 생성  
"splitTest"

클릭

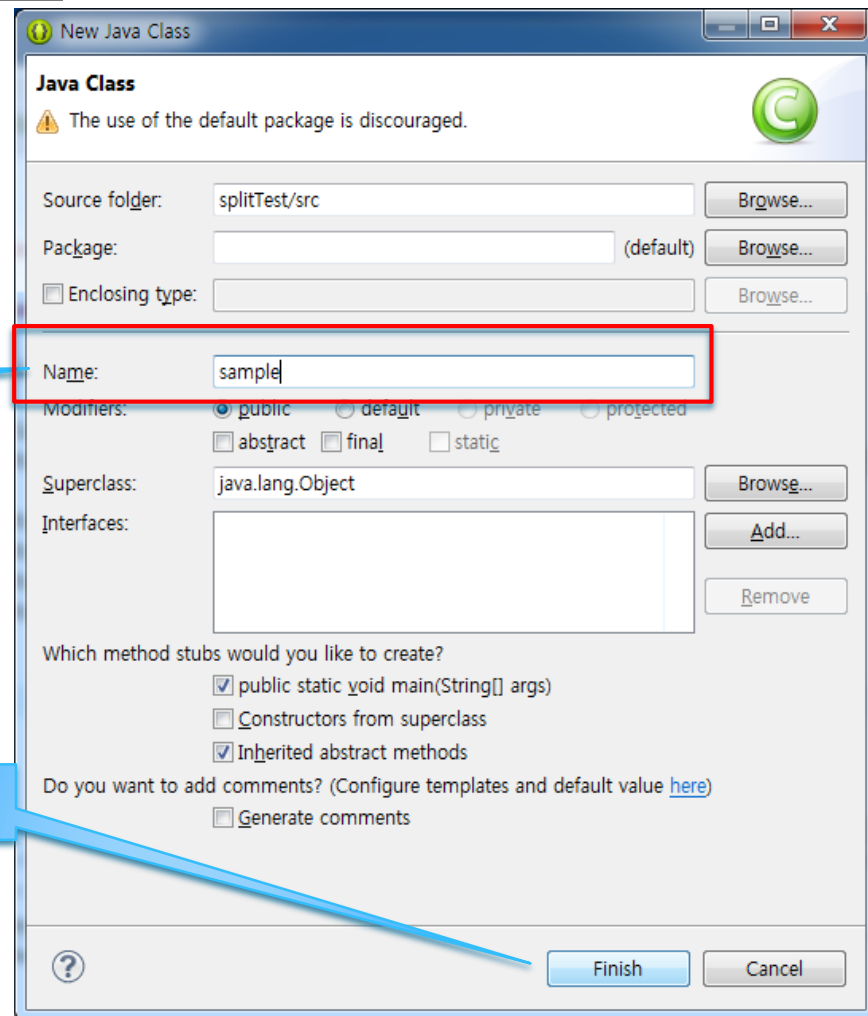




'src' 클릭 후 오른쪽  
마우스 클릭

파일명 입력  
"sample"

클릭



sample.java

```

public class sample {

    public static void main(String[] args) {
        String test = "One and Two and Three and Four";

        String[] array = test.split("and");
        System.out.println(" =>" + array[0] + array[1]+ array[2]+array[3]);

        System.out.println(array[0].length());
        System.out.println(array[1].length());
        System.out.println(array[2].length());
        System.out.println(array[3].length());
    }
}

```

코딩

```

=>One  Two  Three  Four
4
5
7
5

```

```

3
3
5
4

```

글자만 추출하기 위해서는  
어떻게 할 것인가?

```
sample.java ✖  
  
public class sample {  
    public static void main(String[] args) {  
        String test = "One and Two and Three and Four";  
  
        String[] array = test.split(" and ");  
        System.out.println(" =>" + array[0] + array[1] + array[2] + array[3]);  
  
        System.out.println(array[0].length());  
        System.out.println(array[1].length());  
        System.out.println(array[2].length());  
        System.out.println(array[3].length());  
    }  
}
```

공백 " "까지 포함하여 추출

=>OneTwoThreeFour

3

3

5

4

sample.java

```
public class sample {  
  
    public static void main(String[] args) {  
        String test = "문자열 검색을 위한 예제 입니다.";  
  
        if(test.contains("예제")){  
            System.out.println("문자 있어요.");  
        }else{  
            System.out.println("문자 없어요.");  
        }  
  
        if(test.indexOf("예제")>-1){  
            System.out.println("문자 있어요.");  
        }else{  
            System.out.println("문자 없어요.");  
        }  
  
        if(test.matches(".*예제.*")){  
            System.out.println("문자 있어요.");  
        }else{  
            System.out.println("문자 없어요.");  
        }  
    }  
}
```

\*예제\*

문자 있어요.  
문자 있어요.  
문자 있어요.