CSE3040 Java Language

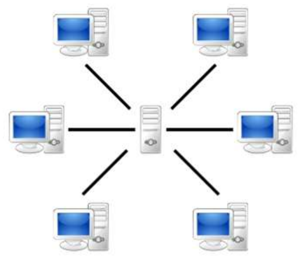
Lecture 21: Networking with Java (1)

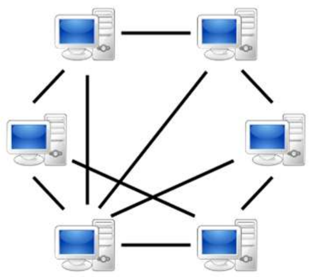
Dept. of Computer Engineering,

Sogang University

Background: Networking Architecture

- Most of today's applications use computer networks.
- Java provides an easy and efficient way for communication between devices through the java.net package.
- Networking architectures
  - Client-Server
    - One server (or possibly a group of servers) provides certain services to many clients.
    - The server waits, and a client makes connection to the server.
    - e.g. mail server, web server (HTTP), file server (FTP), application server
  - Peer-to-Peer (P2P)
    - No distinction between client and server: each device acts as a client and also a server.
    - e.g. bittorrent


client-server

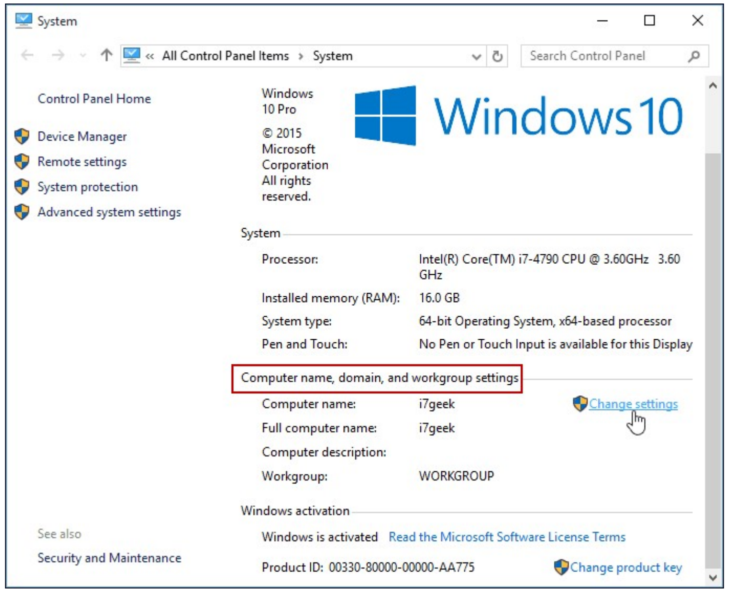

peer-to-peer

Background: IP address

- A device connected to Internet must have a unique IP address in order to communicate with other devices.
- IP address is a 32-bit integer. (IPv4)
  - Since number of devices connecting to the Internet is rapidly increasing, we are running out of IP addresses.
  - Because of that, a new version of IP address IPv6 (128-bit) is designed, but still IPv4 is the commonly used IP address.
- For readability, we write IP address in a dotted decimal notation
  - e.g.) 163.239.1.17
  - Each number ranges from 0 to 255.
- Most IP addresses are unique addresses, but some IP addresses are used for private networks. These are called private IP addresses.
  - 10.0.0.0 – 10.255.255.255
  - 172.16.0.0 – 172.31.255.255
  - 192.168.0.0 – 192.168.255.255
  - Devices having a private IP address need network address translation (NAT) to communicate with devices outside the private network.

Background: Hostname and Domain name

- Hostname
  - a (string) label that is assigned to a device connected to a computer network, used to identify the device.



System

All Control Panel Items › System

Control Panel Home
Device Manager
Remote settings
System protection
Advanced system settings

Windows 10 Pro
© 2015 Microsoft Corporation All rights reserved.

System

Processor:	Intel(R) Core(TM) i7-4790 CPU @ 3.60GHz 3.60 GHz
Installed memory (RAM):	16.0 GB
System type:	64-bit Operating System, x64-based processor
Pen and Touch:	No Pen or Touch Input is available for this Display

Computer name, domain, and workgroup settings

Computer name:	i7geek	Change settings
Full computer name:	i7geek	
Computer description:		
Workgroup:	WORKGROUP	

Windows activation

Windows is activated Read the Microsoft Software License Terms

Product ID: 00330-80000-00000-AA775 Change product key

See also
Security and Maintenance

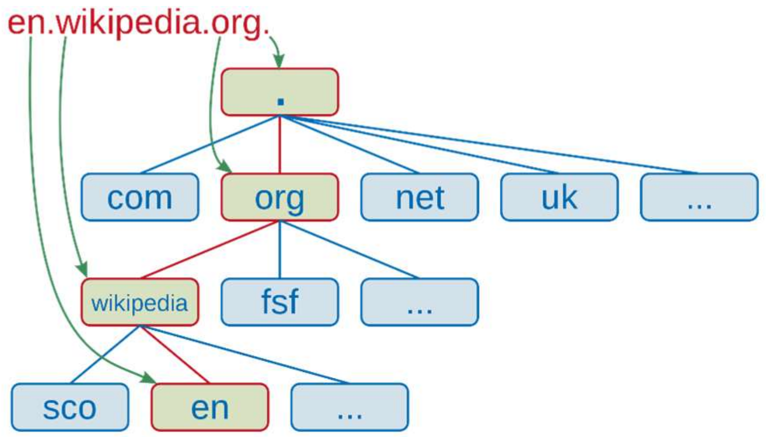
- Domain name
  - An identification string that defines a realm of administrative autonomy within the Internet.
  - A domain name could identify a network domain, or it could represent an Internet Protocol (IP) resource, such as a server computer hosting a web site.
  - e.g.) naver.com, google.com, sogang.ac.kr

서강대학교 SOGANG UNIVERSITY

Background: Hostname and Domain name (cont.)

- Internet Hostname
  - A domain name that is assigned to a host computer.
  - Usually a combination of the host's local name with its parent domain's name.
  - e.g.) www.sogang.ac.kr
    - www is the local hostname
    - sogang.ac.kr is the domain name

- Fully Qualified Domain Name (FQDN)
  - Also called an absolute domain name
  - A domain name that specifies its exact location in the tree hierarchy of the domain name system.
  - e.g.) en.wikipedia.org



Background: Name Server

- A computer application that implements a network service for providing responses to queries against a directory service.

- A name server resides in the Internet, and its IP address is known. (e.g. 168.126.63.1)

- When you type in a URL such as "www.naver.com" in your web browser, the browser must know the IP address that is mapped with "www.naver.com" in order to connect to the server.

- If the browser does not know the IP address, it first sends a query to a name server. The name server will return the IP address to the client host.

- Then, the web browser is able to connect to the server

Background: Port number

- A device is typically assigned a single IP address.

- However, this device may provide multiple services
  - e.g.) A computer may serve as a web server and and an FTP server.

- Since we need a way to distinguish the services, we use port number
  - Port number is a 16-bit integer that ranges from 0 to 65535.
  - e.g.) web server (typically) uses port 80, and ftp server uses port 21.
  - well known ports: http(80), https(443), ftp(21), ssh(22), telnet(23)

- When connecting to a server, the client must specify the IP address and the port number.

InetAddress

- InetAddress is a class offered in Java for handling IP addresses.

- Methods defined in class InetAddress

Method	Description
byte[] getAddress()	Returns the raw IP address of this InetAddress object.
static InetAddress[] getAllByName(String host)	Given the name of a host, returns an array of its IP addresses, based on the configured name service on the system.
static InetAddress getByAddress(byte[] addr)	Returns an InetAddress object given the raw IP address.
static InetAddress getByName(String host)	Determines the IP address of a host, given the host's name.
String getCanonicalHostName()	Gets the fully qualified domain name (FQDN) for this IP address.
String getHostAddress()	Returns the IP address string in textual presentation.
String getHostName()	Gets the host name for this IP address.
static InetAddress getLocalHost()	Returns the address of the local host.
boolean isMulticastAddress()	Utility routine to check if the InetAddress is an IP multicast address.
boolean isLoopbackAddress()	Utility routine to check if the InetAddress is a loopback address.

InetAddress: Example 1

- InetAddress.getByName() retrieves IP address of the given domain name.
  - Since the method throws UnknownHostException which is a checked exception, we should use a try-catch block.
- getHostName() returns the hostname of the InetAddress.
- getHostAddress() returns the IP address of the InetAddress.

```
InetAddress ip = null;

try {
    ip = InetAddress.getByName("www.naver.com");
    System.out.println("getHostName(): " + ip.getHostName());
    System.out.println("getHostAddress(): " + ip.getHostAddress());
    System.out.println("toString(): " + ip.toString());
} catch (UnknownHostException e) {
    e.printStackTrace();
}
```

InetAddress: Example 2

- getAddress() returns the IP address in a byte array.
  - Since byte is a signed type, it ranges from -128 to 127. Should be careful with sign when using the byte array.

```
InetAddress ip = null;

try {
    ip = InetAddress.getByName("www.naver.com");

    byte[] ipAddr = ip.getAddress();
    System.out.println("getAddress(): " + Arrays.toString(ipAddr));

    String result = "";
    for(int i=0; i<ipAddr.length; i++) {
        result += (ipAddr[i] < 0) ? ipAddr[i] + 256 : ipAddr[i];
        result += ".";
    }
    System.out.println("getAddress()+256: " + result);
    System.out.println();

} catch (UnknownHostException e) {
    e.printStackTrace();
}
```

InetAddress: Example 3

- Local host is the host computer where the program is executed.
- If the host computer does not have a domain name, a locally assigned host name will be returned by getHostName().

```
InetAddress ip = null;

try {
    ip = InetAddress.getLocalHost();
    System.out.println("getHostName(): " + ip.getHostName());
    System.out.println("getHostAddress(): " + ip.getHostAddress());
    System.out.println();
} catch (UnknownHostException e) {
    e.printStackTrace();
}
```

InetAddress: Example 4

- An Internet hostname may be mapped to multiple IP addresses.
- getAllByName returns all IP addresses mapped to the given hostname, in an array of InetAddress.

```java
InetAddress[] ipArr = null;

try {
    ipArr = InetAddress.getAllByName("www.naver.com");

    for(int i=0; i<ipArr.length; i++) {
        System.out.println("ipArr["+i+"]: " + ipArr[i]);
    }
} catch (UnknownHostException e) {
    e.printStackTrace();
}
```

URL (Uniform Resource Locator)

- URL is a reference to web resource that specifies its location on a computer network and a mechanism for retrieving it.
- Often called a "web address"
- Web browsers display the URL of a web page in an address bar.
- e.g.) http://www.example.com/index.html
- Format
  - URL = scheme:[//authority]path[?query][#fragment]
  - authority = [userinfo@]host[:port]

  - scheme: a protocol used to access the resource (e.g. http)
  - host: the hostname that holds the resource (e.g. www.example.com)
  - port: port number where the server is listening. (default: http=80, https=443)
  - path: path and filename of the resource
  - query: arguments to the file
  - fragment: index of the fragment within the file

class URL

- Java has class URL which implements methods to handle URLs.

- constructors defined in class URL

Method	Description
URL(String spec)	Creates a URL object from the String representation.
URL(String protocol, String host, String file)	Creates a URL from the specified protocol name, host name, and file name.
URL(String protocol, String host, int port, String file)	Creates a URL object from the specified protocol, host, port number, and file.

- methods for creating a URL object

```
URL url = new URL("http://docs.oracle.com/javase/10/docs/api/java/net/URL.html");
URL url = new URL("http", "docs.oracle.com", "/javase/10/docs/api/java/net/URL.html");
URL url = new URL("http", "docs.oracle.com", 80, "/javase/10/docs/api/java/net/URL.html");
```

class URL

- Methods defined in class URL

Method	Description
String getAuthority()	Gets the authority part of this URL.
Object getContent()	Gets the contents of this URL.
Object getContent(Class<?>[] classes)	Gets the contents of this URL.
int getDefaultPort()	Gets the default port number of the protocol associated with this URL.
String getFile()	Gets the file name of this URL.
String getHost()	Gets the host name of this URL, if applicable.
String getPath()	Gets the path part of this URL.
int getPort()	Gets the port number of this URL.
String getProtocol()	Gets the protocol name of this URL.
String getQuery()	Gets the query part of this URL.
String getRef()	Gets the anchor (also known as the "reference") of this URL.
String getUserInfo()	Gets the userInfo part of this URL.

class URL

- Methods defined in class URL (cont.)

Method	Description
URLConnection openConnection()	Returns a URLConnection instance that represents a connection to the remote object referred to by the URL.
URLConnection openConnection(Proxy proxy)	Same as openConnection(), except that the connection will be made through the specified proxy; Protocol handlers that do not support proxing will ignore the proxy parameter and make a normal connection.
InputStream openStream()	Opens a connection to this URL and returns an InputStream for reading from that connection.
boolean sameFile(URL other)	Compares two URLs, excluding the fragment component.
String toExternalForm()	Constructs a string representation of this URL.
URI toURI()	Returns a URI equivalent of this URL.

URL: Example

- Returns different attributes of a URL object.

```
URL url = new URL("http://www.google.com");
//URL url = new URL("http://mickeymouse@www.google.com");
//URL url = new URL("http://www.youtube.com/results?search_query=java");
//URL url = new URL("https://wikitravel.org/en/Main_Page");
//URL url = new URL("https://en.wikipedia.org/wiki/Java_(programming_language)#Syntax");

System.out.println("url.getAuthority(): " + url.getAuthority());
System.out.println("url.getContent(): " + url.getContent());
System.out.println("url.getDefaultPort(): " + url.getDefaultPort());
System.out.println("url.getPort(): " + url.getPort());
System.out.println("url.getFile(): " + url.getFile());
System.out.println("url.getHost(): " + url.getHost());
System.out.println("url.getPath(): " + url.getPath());
System.out.println("url.getProtocol(): " + url.getProtocol());
System.out.println("url.getQuery(): " + url.getQuery());
System.out.println("url.getRef(): " + url.getRef());
System.out.println("url.getUserInfo(): " + url.getUserInfo());
System.out.println("url.toExternalForm(): " + url.toExternalForm());
System.out.println("url.toURI(): " + url.toURI());
```

Programming Lab #21

21-01. InetAddress Example 1

- Execute the following code and understand the result.
- Try different servers such as www.google.com or www.sogang.ac.kr.

```
public class Ex21_01 {
    public static void main(String[] args) {
        InetAddress ip = null;

        try {
            ip = InetAddress.getByName("www.naver.com");
            System.out.println("getHostName(): " + ip.getHostName());
            System.out.println("getHostAddress(): " + ip.getHostAddress());
            System.out.println("toString(): " + ip.toString());
        } catch (UnknownHostException e) {
            e.printStackTrace();
        }
    }
}
```

21-02. InetAddress Example 2

- Execute the following code and understand the result.

```
public class Ex21_02 {
	public static void main(String[] args) {
		InetAddress ip = null;

		try {
				ip = InetAddress.getByName("www.naver.com");

				byte[] ipAddr = ip.getAddress();
				System.out.println("getAddress(): " + Arrays.toString(ipAddr));

				String result = "";
				for(int i=0; i<ipAddr.length; i++) {
						result += (ipAddr[i] < 0) ? ipAddr[i] + 256 : ipAddr[i];
						result += ".";
				}

				System.out.println("getAddress()+256: " + result);
				System.out.println();
		}
		catch(UnknownHostException e) {
				e.printStackTrace();
		}
	}
}
```

21-03. InetAddress Example 3

- Execute the following code and understand the result.

```
public class Ex21_03 {
    public static void main(String[] args) {
        InetAddress ip = null;

        try {
                ip = InetAddress.getLocalHost();
                System.out.println("getHostName(): " + ip.getHostName());
                System.out.println("getHostAddress(): " + ip.getHostAddress());
                System.out.println();
        }
        catch(UnknownHostException e) {
                e.printStackTrace();
        }
    }
}
```

서강대학교 SOGANG UNIVERSITY

21-04. InetAddress Example 4

- Execute the following code and understand the result.
- Try different URLs including the ones commented out in the code.

```
public class Ex21_04 {
	public static void main(String[] args) throws Exception {
		URL url = new URL("http://www.google.com");
		//URL url = new URL("http://mickeymouse@www.google.com");
		//URL url = new URL("http://www.youtube.com/results?search_query=java");
		//URL url = new URL("https://wikitravel.org/en/Main_Page");
		//URL url = new URL("https://en.wikipedia.org/wiki/Java_(programming_language)#Syntax");

		System.out.println("url.getAuthority(): " + url.getAuthority());
		System.out.println("url.getContent(): " + url.getContent());
		System.out.println("url.getDefaultPort(): " + url.getDefaultPort());
		System.out.println("url.getPort(): " + url.getPort());
		System.out.println("url.getFile(): " + url.getFile());
		System.out.println("url.getHost(): " + url.getHost());
		System.out.println("url.getPath(): " + url.getPath());
		System.out.println("url.getProtocol(): " + url.getProtocol());
		System.out.println("url.getQuery(): " + url.getQuery());
		System.out.println("url.getRef(): " + url.getRef());
		System.out.println("url.getUserInfo(): " + url.getUserInfo());
		System.out.println("url.toExternalForm(): " + url.toExternalForm());
		System.out.println("url.toURI(): " + url.toURI());
	}
}
```

End of Class

Instructor office: AS818A

Email: jso1@sogang.ac.kr