|  |  |
| --- | --- |
| H:\LOGO-NXV\Hai__090908__02_1_den.jpg | Faculty of Information Technology  H A N O I U N I V E R S I T Y |

61FIT3NPR – Network Programming

Install Java and URL/URI in Java

# A. INSTALL JAVA

**I. INSTALL INTELLIJ IDEA**

If you don’t want to use Eclipse, you can choose another IDE like IntelliJ IDEA.

Step 1: Download IntelliJ IDEA

<https://www.jetbrains.com/idea/download/?section=windows>

You should choose Community edition.

Step 2: Extract and install IntelliJ IDEA

**II. USING ECLIPSE**

**1. Download JDK, Eclipse**

**1.1. JDK:**

JDK is Java Development Kit which include JRE (Java Runtime Environment). Choose version that is compatible to your OS.

<http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html>

**1.2. Eclipse IDE for Java EE Developers:**

Choose version that is compatible to your OS and JDK downloaded above. For lab computer, choose Windows 32 bit file.

<https://www.eclipse.org/downloads/packages/release/photon/r/eclipse-ide-java-ee-developers>

**2. Install JDK, Eclipse**

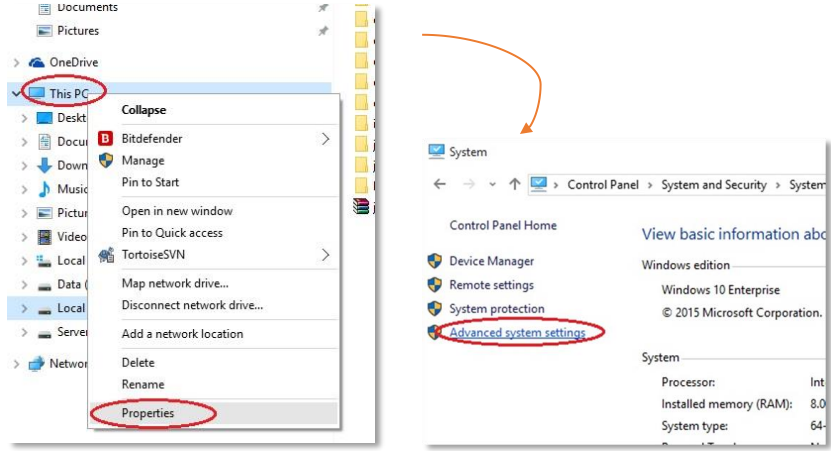
**2.1. Install JDK:**

Step 1: Run jdk.exe file or Extract jdk.rar file to a folder on your hard drive, for example,

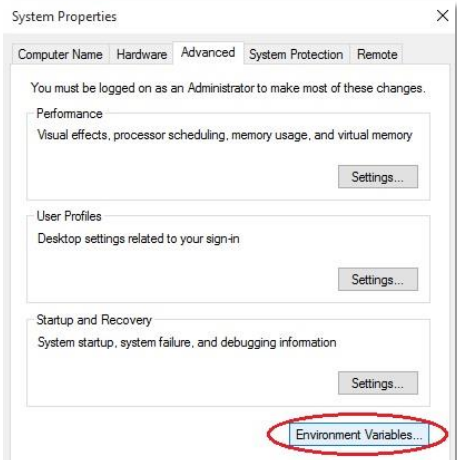
Step 2: Set environment variable:

In Windows, right click on This PC -> choose Properties -> choose Advanced system settings.

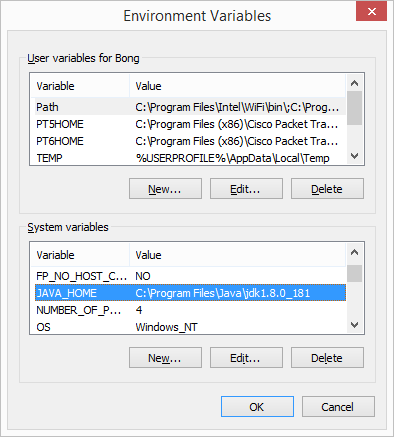
or go to Control Panel -> System and Security -> System -> choose Advanced system settings.



In System Properties dialog, choose Environment variables.



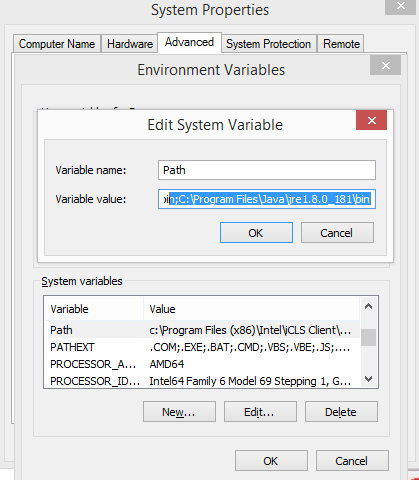
In System variables, Add new variable (or edit if already exist one) with Variable name JAVA\_HOME and Variable value is your path to jdk folder.



In System variables, also go to edit variable PATH, add

C:\Program Files\Java\jdk1.8.0\_181\bin;C:\Program Files\Java\jre1.8.0\_181\bin

at the end of Variable value.



**2.2. Install Eclipse:**

Step 1: Extract the zip file to a folder on your hard drive, for example, E:\Java

Step 2: Go to the folder you have extracted and run eclipse.exe

**B. FIRST PROGRAM**

1. Parsing an URL

String getProtocol()

String getHost()

int getPort()

String getFile()

String getRef()

**import** java.net.\* ;

**public** **class** ParseURL {

**public** **static** **void** main(String[]args) **throws** MalformedURLException {

URL url = **new** URL("http://java.sun.com:80/docs/books/tutorial/intro.html#DOWNLOADING") ;

System.***out***.println("Protocol = "+url.getProtocol()) ;

System.***out***.println("Host = "+url.getHost()) ;

System.***out***.println("FileName = "+url.getFile()) ;

System.***out***.println("Port = "+url.getPort()) ;

System.***out***.println("Reference= "+url.getRef()) ;

}

}

1. Reading directly from a URL using openStream() method that returns an InputStream

Here is an example that reads a URL directly. It actually reads from the Internet and displays it line by line to the console. Notice that it reads the file as a text file, so we simply get the HTML code. Also, you must be connected to the internet to run this code:

**Example:**

**import** java.net.\* ;

**import** java.io.\* ;

**public** **class** URLReader {

**public** **static** **void** main(String[] args) **throws** Exception {

URL url = **new** URL("https://www.hanu.vn") ;

BufferedReader br = **new** BufferedReader (**new** InputStreamReader(url.openStream())) ;

String line ;

**while** ((line=br.readLine())!=**null**)

System.***out***.println(line) ;

br.close() ;

}

}

1. Reading from a URL connection:

**Example:**

**import** java.net.\* ;

**import** java.io.\* ;

**public** **class** URLConnectionReader {

**public** **static** **void** main(String[] args) **throws** Exception {

URL url = **new** URL("https://www.hanu.vn") ;

URLConnection uc = url.openConnection() ;

BufferedReader br=**new** BufferedReader(**new** InputStreamReader(uc.getInputStream()));

String line ;

**while** ((line=br.readLine())!=**null**)

System.***out***.println(line) ;

br.close() ;

}

}

Here is a modification to the above example that reads the URL by making a **URLConnection** first. Since the tasks of opening a connection to a webpage and reading the contents may both generate an **IOException**, we cannot distinguish the kind of error that occurred. By trying to establish the connection first, if any **IOExceptions** occur, we know they are due to a connection problem. Once the connection has been established, then any further **IOException** errors would be due to the reading of the webpage data

**import** java.net.\*;

**import** java.io.\*;

**public** **class** URLConnectionReader2 {

**public** **static** **void** main(String[] args) {

URL url = **null**;

BufferedReader in = **null**;

**try** {

url = **new** URL("https://en.wikipedia.org/wiki/Computer\_science");

} **catch**(MalformedURLException e) {

System.***out***.println("Cannot find webpage " + url); System.*exit*(-1);

}

**try** {

URLConnection aConnection = url.openConnection();

in = **new** BufferedReader(

**new** InputStreamReader(aConnection.getInputStream()));

}

**catch** (IOException e) {

System.***out***.println("Cannot connect to webpage " + url); System.*exit*(-1);

}

**try** {

// Now read the webpage file

String lineOfWebPage;

**while** ((lineOfWebPage = in.readLine()) != **null**) System.***out***.println(lineOfWebPage);

in.close(); // Close the connection to the net

} **catch**(IOException e) {

System.***out***.println("Cannot read from webpage " + url);

}

}

}