



# MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)  
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu



LECTURE HANDOUTS

L 01

MCA

I / II

Course Code&Name : 19CAB11 & INTERNET AND JAVA PROGRAMMING

Name of the Faculty : Mrs.G.Krishnaveni

Year / Semester/Section : I / II

Date of Lecture: 22.02.2021

## Topic of Lecture: Domain Name System

### Introduction : ( Maximum 5 sentences)

- The Domain Name System (DNS) turns domain names into IP addresses, which browsers use to load internet pages.
- Every device connected to the internet has its own IP address, which is used by other devices to locate the device.

### Prerequisite knowledge for Complete understanding and learning of Topic: ( Max. Four important topics)

- Basics of Internet
- Introduction of E-mail

### Detailed content of the Lecture:

#### Domain Name System

- The Domain Name System (DNS) translates human-readable domain names to machine-readable IP addresses.
- A DNS name server stores the DNS records for a zone, and responds with answers to queries against its database.
- When you type a domain name into your browser, your operating system queries several DNS name servers until it finds the authoritative name server for that domain.
- The authoritative name server then responds with an IP address or other requested record data. The answer is then relayed back to your browser and the DNS record is resolved to the web page.

#### DNS use cases and benefits

#### Intelligent traffic routing

- DNS traffic management allows customers to intelligently route user traffic across the internet by offering geolocation steering, load balancing, ASN, and IP-prefix steering.
- This provides optimized response times for web-facing applications, allowing users to reach them as quickly as possible.

- This service is integrated with Oracle Health check services for endpoint monitoring.

### **Easy-to-use APIs**

- Customers streamline DNS integration with web-facing applications and automate routine DNS configuration tasks using the console, OCI API, terraform and multiple SDKs.

### **DNS Work**

#### **DNS recursor:**

- The DNS recursor, which is also referred to as a DNS resolver, receives the query from the DNS client.
- Then it communicates with other DNS servers to find the right IP address.
- As it does this, it makes queries that get sent to the other three DNS servers: root nameservers, top-level domain (TLD) nameservers, and authoritative nameservers.

#### **Root nameservers:**

- The root nameserver is designated for the internet's DNS root zone.
- Its job is to answer requests sent to it for records in the root zone.

#### **TLD nameservers:**

- A TLD nameserver keeps the IP address of the second-level domain contained within the TLD name.
- It then releases the website's IP address and sends the query to the domain's nameserver.

#### **Authoritative nameservers:**

- An authoritative nameserver is what gives you the real answer to your DNS query.
- There are two types of authoritative nameservers: a master server or primary nameserver and a slave server or secondary nameserver.
- The master server keeps the original copies of the zone records, while the slave server is an exact copy of the master server.

### **Video Content / Details of website for further learning (if any):**

- <https://www.oracle.com/cloud/networking/dns>
- <https://www.fortinet.com/resources/cyberglossary/what-is-dns>

### **Important Books/Journals for further learning including the page nos.:**

- **Net Reference**

**Course Faculty**

**Verified by HOD**



# MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)  
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu



LECTURE HANDOUTS

L 02

MCA

I / II

Course Code &Name : 19CAB11 & INTERNET AND JAVA PROGRAMMING

Name of the Faculty : Mrs.G.Krishnaveni

Year / Semester/Section : I / II

Date of Lecture: 23.02.2021

## Topic of Lecture: Exchanging E-mail

### Introduction : ( Maximum 5 sentences)

- **Electronic mail (email or e-mail)** is a method of exchanging messages ("mail") between people using electronic devices.
- Email operates across [computer networks](#), primarily the [Internet](#). Today's email systems are based on a [store-and-forward](#) model.

### Prerequisite knowledge for Complete understanding and learning of Topic: ( Max. Four important topics)

- Basics of Internet
- Introduction of E-mail

### Detailed content of the Lecture:

- E-mail is defined as **the transmission of messages on the Internet**.
- It is one of the most commonly used features over communications networks that may contain text, files, images, or other attachments.

### E-Mail Address

- Each user of email is assigned a unique name for his email account.
- This name is known as E-mail address. Different users can send and receive messages according to the e-mail address.
- E-mail is generally of the form username@domainname.
- For example, webmaster@tutorialspoint.com is an e-mail address where webmaster is username and tutorialspoint.com is domain name.
- The username and the domain name are separated by @ (**at**) symbol.
- E-mail addresses are not case sensitive.
- Spaces are not allowed in e-mail address.

### E-mail Message Components

E-mail message comprises of different components: E-mail Header, Greeting, Text, and Signature. These components are described in the following diagram:



The first five lines of an E-mail message is called E-mail header. The header part comprises of following fields:

- From
- Date
- To
- Subject
- CC
- BCC

### From

The **From** field indicates the sender's address i.e. who sent the e-mail.

### Date

The **Date** field indicates the date when the e-mail was sent.

### To

The **To** field indicates the recipient's address i.e. to whom the e-mail is sent.

### Subject

The **Subject** field indicates the purpose of e-mail. It should be precise and to the point.

### CC

**CC** stands for Carbon copy. It includes those recipient addresses whom we want to keep informed but not exactly the intended recipient.

### BCC

**BCC** stands for Black Carbon Copy. It is used when we do not want one or more of the recipients to know that someone else was copied on the message.

### Greeting

Greeting is the opening of the actual message. Eg. Hi Sir or Hi Guys etc.

**Text**

It represents the actual content of the message.

**Signature**

This is the final part of an e-mail message. It includes Name of Sender, Address, and Contact Number.

**Advantages**

E-mail has proved to be powerful and reliable medium of communication. Here are the benefits of **E-mail**:

- Reliable
- Convenience
- Speed
- Inexpensive
- Printable
- Global
- Generality

**Reliable**

Many of the mail systems notify the sender if e-mail message was undeliverable.

**Convenience**

There is no requirement of stationary and stamps. One does not have to go to post office. But all these things are not required for sending or receiving an mail.

**Speed**

E-mail is very fast. However, the speed also depends upon the underlying network.

**Inexpensive**

The cost of sending e-mail is very low.

**Printable**

It is easy to obtain a hardcopy of an e-mail. Also an electronic copy of an e-mail can also be saved for records.

**Global**

E-mail can be sent and received by a person sitting across the globe.

**Generality**

It is also possible to send graphics, programs and sounds with an e-mail.

**Disadvantages**

Apart from several benefits of E-mail, there also exists some disadvantages as discussed below:

- Forgery
- Overload
- Misdirection
- Junk
- No response

### **Forgery**

E-mail doesn't prevent from forgery, that is, someone impersonating the sender, since sender is usually not authenticated in any way.

### **Overload**

Convenience of E-mail may result in a flood of mail.

### **Misdirection**

It is possible that you may send e-mail to an unintended recipient.

### **Junk**

Junk emails are undesirable and inappropriate emails. Junk emails are sometimes referred to as spam.

### **No Response**

It may be frustrating when the recipient does not read the e-mail and respond on a regular basis.

### **Video Content / Details of website for further learning (if any):**

- <https://en.wikipedia.org/wiki/Email>
- [https://www.tutorialspoint.com/internet\\_technologies/e\\_mail\\_overview.htm](https://www.tutorialspoint.com/internet_technologies/e_mail_overview.htm)
- [https://www.mailenable.com/documentation/10.0/Standard/How\\_Internet\\_Email\\_Works.html](https://www.mailenable.com/documentation/10.0/Standard/How_Internet_Email_Works.html)
- 

### **Important Books/Journals for further learning including the page nos.:**

- Net Reference

**Course Faculty**

**Verified by HOD**



# MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)  
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu



LECTURE HANDOUTS

L 03

MCA

I / II

Course Code &Name : 19CAB11 & INTERNET AND JAVA PROGRAMMING

Name of the Faculty : Mrs. G. Krishnaveni

Year / Semester/Section : I / II

Date of Lecture: 25.02.2021

## Topic of Lecture: Sending and Receiving Files Fighting Spam

### Introduction : ( Maximum 5 sentences)

- Email spam, also known as junk email, refers to unsolicited email messages, usually sent in bulk to a large list of recipients.
- Spam is any kind of unwanted, unsolicited digital communication that gets sent out in bulk. Often spam is sent via email, but it can also be distributed via text messages, phone calls, or social media.

### Prerequisite knowledge for Complete understanding and learning of Topic:

#### ( Max. Four important topics)

- Basics of Internet
- Introduction of E-mail

### Detailed content of the Lecture:

- Spam is not an acronym for a computer threat, although some have been proposed (stupid pointless annoying malware, for instance).
- The inspiration for using the term “spam” to describe mass unwanted messages is a Monty Python skit in which the actors declare that everyone must eat the food Spam, whether they want it or not.

### Types of spam

- Spammers use many forms of communication to bulk-send their unwanted messages.
- Some of these are marketing messages peddling unsolicited goods.
- Other types of spam messages can spread malware, trick you into divulging personal information, or scare you into thinking you need to pay to get out of trouble.
- Email spam filters catch many of these types of messages, and phone carriers often warn you of a “spam risk” from unknown callers.

### Phishing emails

- Phishing emails are a type of spam cybercriminals send to many people, hoping to “hook” a few people.
- Phishing emails trick victims into giving up sensitive information like website logins or credit card information.
- Adam Kujawa, Director of Malwarebytes Labs, says of phishing emails: “Phishing is the simplest kind of cyberattack and, at the same time, the most dangerous and effective.

- That is because it attacks the most vulnerable and powerful computer on the planet: the human mind.”

### **Email spoofing**

- Spoofed emails mimic, or spoof, an email from a legitimate sender, and ask you to take some sort of action.
- Well-executed spoofs will contain familiar branding and content, often from a large well-known company such as PayPal or Apple.
- Common email spoofing spam messages include:
  - A request for payment of an outstanding invoice
  - A request to reset your password or verify your account
  - Verification of purchases you didn't make
  - Request for updated billing information

### **Tech support scams**

- In a tech support scam, the spam message indicates that you have a technical problem and you should contact tech support by calling the phone number or clicking a link in the message.
- Like email spoofing, these types of spam often say they are from a large technology company like Microsoft or a cyber security company like Malware bytes.
- If you think you have a technical issue or malware on your computer, tablet, or smart phone, you should always go to the official website of the company you want to call for tech support to find the legitimate contact information.
- Remote tech support often involves remote access to your computer to help you, and you don't want to accidentally give that access to a tech support scammer.

### **Current event scams**

- Hot topics in the news can be used in spam messages to get your attention.
- In 2020 when the world was facing the Covid-19 pandemic and there was an increase in work-from-home jobs, some scammers sent spam messages promising remote jobs that paid in Bitcoin.
- During the same year, another popular spam topic was related to offering financial relief for small businesses, but the scammers ultimately asked for bank account details.
- News headlines can be catchy, but beware of them in regards to potential spam messages.

### **Advance-fee scams**

- This type of spam is likely familiar to anyone who has been using email since the 90s or 2000s.
- Sometimes called “Nigerian prince” emails as that was the purported message sender for many years, this type of spam promises a financial reward if you first provide a cash advance.
- The sender typically indicates that this cash advance is some sort of processing fee or earnest money to unlock the larger sum, but once you pay, they disappear.
- To make it more personal, a similar type of scam involves the sender pretending to be a family member that is in trouble and needs money, but if you pay, unfortunately the outcome is the same.



## **Malspam**

- Short for “malware spam” or “malicious spam,” malspam is a spam message that delivers malware to your device.
- Unsuspecting readers who click on a link or open an email attachment end up with some type of malware including ransomware, Trojans, bots, info-stealers, cryptominers, spyware, and keyloggers.
- A common delivery method is to include malicious scripts in an attachment of a familiar type like a Word document, PDF file, or PowerPoint presentation.
- Once the attachment is opened, the scripts run and retrieve the malware payload.

## **Spam calls and spam texts**

- Have you ever received a robocall? That’s call spam.
- A text message from an unknown sender urging you to click an unknown link?
- That’s referred to as text message spam or “smishing,” a combination of SMS and phishing.
- If you’re receiving spam calls and texts on your Android or iPhone, most major carriers give you an option to report spam.
- Blocking numbers is another way to combat mobile spam.
- In the US, you can add your phone number to the National Do Not Call Registry to try to cut down on the amount of unwanted sales calls you receive, but you should still be alert to scammers who ignore the list.

## **How can I stop spam?**

- While it may not be possible to avoid spam altogether, there are steps you can take to help protect yourself against falling for a scam or getting phished from a spam message:

## **Learn to spot phishing**

- All of us can fall victim to phishing attacks.
- We may be in a rush and click a malicious link without realizing.
- If a new type of phishing attack comes out, we may not readily recognize it.
- To protect yourself, learn to check for some key signs that a spam message isn’t just annoying—it’s a phishing attempt:

### **1. Sender’s email address:**

- If an email from a company is legitimate, the sender’s email address should match the domain for the company they claim to represent.
- Sometimes these are obvious, like example@abkljzr09348.biz, but other times the changes are less noticeable, like example@paypal.com instead of paypal.com.

### **2. Missing personal information:**

- If you are a customer, the company should have your information and will likely address you by your first name.
- A missing personal greeting alone isn’t enough to spot a phishing email, but it’s one thing to look for, especially in messages that say they are from a company with whom you do business.
- Receiving an email that says your account has been locked or you owe money is cause to worry, and sometimes we rush to click a link in order to fix the problem.

- If it's phishing, that's exactly what the sender wants, so be careful and check if the email is generic or addressed specifically to you.

### 3. **Links:**

- Beware of all links, including buttons in an email.
- If you get a message from a company with whom you have an account, it's wise to log in to your account to see if there is a message there rather than just clicking the link in the message without verifying first.
- You can contact the company to ask if a suspicious message is legitimate or not. If you have any doubts about a message, don't click any links.

### 4. **Grammatical errors:**

- We all make them, but a company sending out legitimate messages probably won't have a lot of punctuation errors, poor grammar, and spelling mistakes.
- These can be another red flag to indicate that the email could be suspect.

### 5. **Too-good-to-be-true offers:**

- Many phishing messages pretend to be from large, well-known companies, hoping to ensnare readers who happen to do business with the company.
- Other phishing attempts offer something for free like cash or a desirable prize.
- The saying is often true that if something sounds too good to be true it probably is, and this can be a warning that a spam message is trying to get something from you, rather than give you something.

### 6. **Attachments:**

- Unless you are expecting an email with attachments, always be wary before opening or downloading them.
- Using anti-malware software can help by scanning files that you download for malware.

## **Report spam**

- Email providers have gotten pretty good at filtering out spam, but when messages make it through to your inbox, you can report them.
- This is true for spam calls and text messages, as many carriers give you the ability to report spam as well. You can also choose to block the sender, often in the same step as reporting the message.
- Reporting spam can help your email provider or phone service carrier get better at detecting spam.
- If legitimate emails get sent to your spam filter, you can report that they should not be marked as spam, and that also provides useful information on what should not be filtered.
- Another helpful step is to add senders you want to hear from to your contacts list proactively.

## **Use two factor-authentication (2FA)**

- With two-factor or multi-factor authentication, even if your username and password are compromised via a phishing attack, cybercriminals won't be able to get around the additional authentication requirements tied to your account.
- Additional authentication factors include secret questions or verification codes sent to your phone via text message.

**Install cyber security**

- In the event that you click a bad link or download malware sent to you via spam, good cyber security software will recognize the malware and shut it down before it can do any damage to your system or network.
- With products for home and business, Malware bytes has got you covered wherever technology takes you.

**Video Content / Details of website for further learning (if any):**

- <https://www.cisa.gov/uscert/publications/virus-basics>
- <https://msatechnosoft.in/blog/avoid-e-mail-viruses-email-security>

**Important Books/Journals for further learning including the page nos.:**

- Net Reference

**Course Faculty**

**Verified by HOD**



# MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)  
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu



LECTURE HANDOUTS

L 04

MCA

I / II

Course Code & Name : 19CAB11 & INTERNET AND JAVA PROGRAMMING

Name of the Faculty : Mrs. G. Krishnaveni

Year / Semester/Section : I / II

Date of Lecture: 25.02.2021

## Topic of Lecture: Sorting Mail and avoiding e-mail viruses

### Introduction : ( Maximum 5 sentences)

- A computer virus is a program that spreads by first infecting files or the system areas of a computer or network router's hard drive and then making copies of itself.
- Some viruses are harmless, others may damage data files, and some may destroy files.

### Prerequisite knowledge for Complete understanding and learning of Topic: ( Max. Four important topics)

- Basics of Internet
- Introduction of E-mail

### Detailed content of the Lecture:

- Viruses used to be spread when people shared floppy disks and other portable media, now viruses are primarily spread through email messages.
- Unlike worms, viruses often require some sort of user action (e.g., opening an email attachment or visiting a malicious web page) to spread.
- A virus is simply a computer program--it can do anything that any other program you run on your computer can do. Some viruses are designed to deliberately damage files, and others may just spread to other computers.
- A worm is a type of virus that can spread without human interaction. Worms often spread from computer to computer and take up valuable memory and network bandwidth, which can cause a computer to stop responding. Worms can also allow attackers to gain access to your computer remotely.
- A Trojan horse is a computer program that is hiding a virus or other potentially damaging program. A Trojan horse can be a program that purports to do one action when, in fact, it is performing a malicious action on your computer. Trojan horses can be included in software that you download for free or as attachments in email messages.

### Can I get a virus by reading my email messages?

- Most viruses, Trojan horses, and worms are activated when you open an attachment or click a link contained in an email message.
- If your email client allows scripting, then it is possible to get a virus by simply opening a message. It's best to limit what HTML is available in your email messages. The safest way to view email messages is in plain text.

### How can I avoid a virus infection from email?

- Most users get viruses from opening and running unknown email attachments.

- Never open anything that is attached to an email message unless you know the contents of the file.
- If you receive an attachment from a familiar email address, but were not expecting anything, you should contact the sender before opening the attachment.
- If you receive a message with an attachment and you do not recognize the sender, you should delete the message.

#### **What are some tips to avoid viruses and lessen their impact?**

- Install anti-virus software from a reputable vendor. Update it and use it regularly.
- In addition to scanning for viruses on a regular basis, install an "on access" scanner (included in most anti-virus software packages) and configure it to start each time you start up your computer. This will protect your system by checking for viruses each time you run an executable file.
- Use a virus scan before you open any new programs or files that may contain executable code. This includes packaged software that you buy from the store as well as any program you might download from the Internet.
- If you are a member of an online community or chat room, be very careful about accepting files or clicking links that you find or that people send you within the community.
- Make sure you back up your data (documents, bookmark files, important email messages, etc.) on disc so that in the event of a virus infection, you do not lose valuable work.

#### **Anti-virus software**

- **Automatic scans** – Most anti-virus software can be configured to automatically scan specific files or directories in real time and prompt you at set intervals to perform complete scans.
- **Manual scans** – If your anti-virus software does not automatically scan new files, you should manually scan files and media you receive from an outside source before opening them. This process includes:
  - Saving and scanning email attachments or web downloads rather than opening them directly from the source.
  - Scanning media, including CDs and DVDs, for malware before opening files.

#### **Video Content / Details of website for further learning (if any):**

- <https://www.cisa.gov/uscert/publications/virus-basics>
- <https://msatechnosoft.in/blog/avoid-e-mail-viruses-email-security>

#### **Important Books/Journals for further learning including the page nos.:**

- **Net Reference**

**Course Faculty**

**Verified by HOD**



# MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)  
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu



LECTURE HANDOUTS

L 05

MCA

I / II

Course Code &Name : 19CAB11 & INTERNET AND JAVA PROGRAMMING

Name of the Faculty : Mrs. G. Krishnaveni

Year / Semester/Section : I / II

Date of Lecture: 26.02.2021

## Topic of Lecture: Chatting and Conferencing on the Internet

### Introduction : ( Maximum 5 sentences)

- Chatting refers to the kind of communication done with the help of the internet which present live transmission of text messages from sender to receiver.
- Online chatting can be termed as the point-to-point, one sender-to-one receiver, or one sender-to-many receiver. It also features voice, video, and also web conferencing services.

### Prerequisite knowledge for Complete understanding and learning of Topic: ( Max. Four important topics)

- Basics of Internet
- Introduction of E-mail

Detailed content of the Lecture:

**Video conferencing** or **Video teleconferencing** is a method of communicating by two-way video and audio transmission with help of telecommunication technologies.

Modes of Video Conferencing

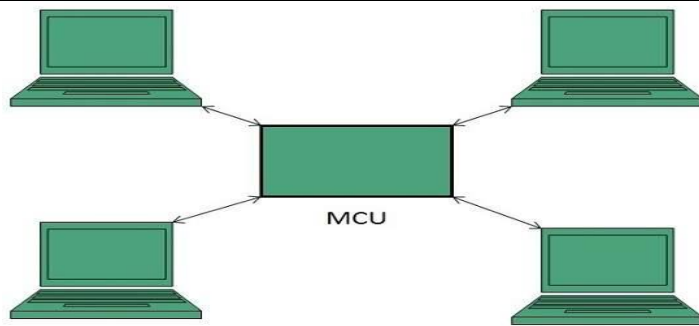
Point-to-Point

This mode of conferencing connects two locations only.



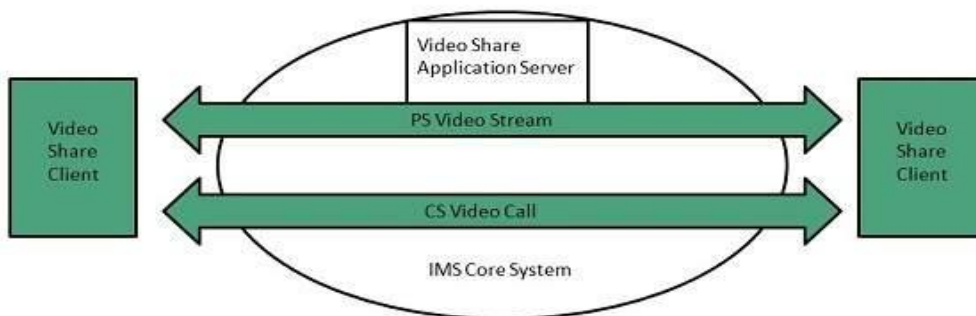
Multi-point

This mode of conferencing connects more than two locations through **Multi-point Control Unit (MCU)**.



## Video Sharing

- Video sharing is an IP Multimedia System (IMS) service that allows user to switch voice calls to unidirectional video streaming session.
- The video streaming session can be initiated by any of the parties. Moreover, the video source can be the camera or the pre-recorded video clip.
- Internet chatting involves real-time instant text messaging between two or more users in chat rooms. They are many different types of Internet chats all with different purposes.



## Benefits

- Internet chat rooms allow you to communicate with different kinds of people from all over the world.
- They allow you to meet different kinds of people who share similar interests, goals, hobbies and desires.
- Internet chats can also be a great learning center ( e.g. chatting forums) where people can ask questions and receive answers on products and services, computer troubleshooting and more.

## Features

- Internet chats not only allow you to send and receive instant messages, they also allow you to share pictures, and files.
- Some Internet chat rooms include emoticons which are smiley faces used to describe what your present emotion is. Some Internet chats include sound effects which range from serious to silly and allow you to. Other chat rooms allow you to change color combinations to create a

theme or background that works for you as you are chatting.

- There are three commonly used types of chat. They are **Instant Messaging, ICQ, and IRC**. Instant messaging (IM) is one of the most popular forms of chat.

**Video Content / Details of website for further learning (if any):**

- <https://slidetodoc.com/online-chatting-and-conferencing-concepts-introduction-a-group/>
- <https://www.techwalla.com/articles/definition-of-internet-chat>
- <https://brainly.in/question/31782263>
- 

**Important Books/Journals for further learning including the page nos.:**

- **Net Reference**

**Course Faculty**

**Verified by HOD**





# MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)  
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu



LECTURE HANDOUTS

L 06

MCA

I / II

Course Code &Name : 19CAB11 & INTERNET AND JAVA PROGRAMMING

Name of the Faculty : Mrs. G. Krishnaveni

Year / Semester/Section : I / II

Date of Lecture: 01.03.2021

## Topic of Lecture: Online Chatting , Messaging

### Introduction : ( Maximum 5 sentences)

- On the Internet, chatting is **talking to other people who are using the Internet at the same time you are.**
- Usually, this "talking" is the exchange of typed-in messages requiring one site as the repository for the messages (or "chat site") and a group of users who take part from anywhere on the Internet..

### Prerequisite knowledge for Complete understanding and learning of Topic:

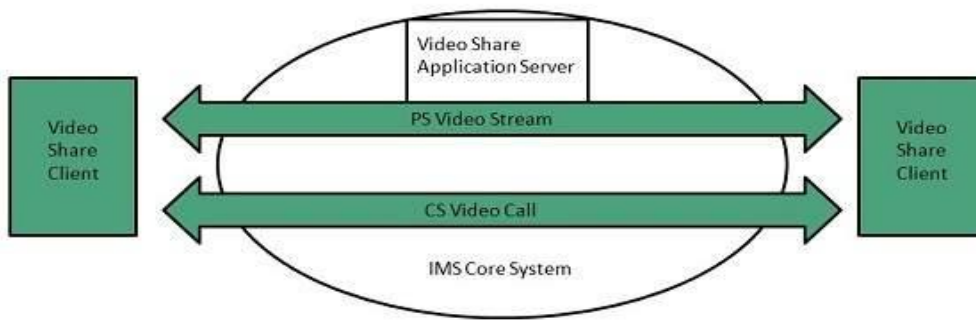
- Basics of Internet
- Introduction of E-mail

### Detailed content of the Lecture:

- Chatting on the Internet comes in many forms. You can have one-on-one chatting via instant messaging software.
- You can also have group discussions via a chat room or a forum. However, as with anything, you can find advantages and disadvantages to having Internet chats.

### Multitasking

- We can complete other tasks while chatting over the Internet.
- For example, you can read and reply to emails, finish typing a document, or have a conversation with someone who is in the room.
- This can be an advantage if you are busy and don't have much time for personal chatting. This can also be a disadvantage because you aren't giving the other person or task your full attention. Time Management
- Because you can multitask while chatting on the Internet, it can become easy to lose track of time. You can end up chatting longer than you intended to, which can make it difficult to complete other tasks.



### Communication Barriers

- The intent of someone's remarks can be hard to determine over the Internet. It's easy for you to seem offensive when chatting on the Internet because the person on the other end cannot see your face or hear your tone of voice.
- Therefore, you have to be very careful with the way you word things.

### Dangers

- If you chat with strangers over the Internet, you can open yourself to predators.
- People can lie about who they are in an effort to hurt you in some way.
- Your conversations can also be saved, which can come back to haunt you if someone has bad intentions.
- According to the United States Computer Emergency Readiness Team, you should be careful about what you reveal unless you are certain of who the person on the other end of the chat is.
- The US-CERT also recommends updating your security settings to protect yourself from a software attack from a malicious user.

### Video Content / Details of website for further learning (if any):

- [https://en.wikipedia.org/wiki/Online\\_chat](https://en.wikipedia.org/wiki/Online_chat)
- <https://www.geeksforgeeks.org/what-is-chatting-definition-types-platforms-risks/>

### Important Books/Journals for further learning including the page nos.:

- Net Reference

Course Faculty

Verified by HOD



# MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)  
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu



LECTURE HANDOUTS

L 07

MCA

I / II

**Course Code &Name : 19CAB11 & INTERNET AND JAVA PROGRAMMING**

**Name of the Faculty : Mrs. G. Krishnaveni**

**Year / Semester/Section : I / II**

**Date of Lecture: 02.03.2021**

**Topic of Lecture: Voice and Video Conferencing**

**Introduction : ( Maximum 5 sentences)**

- Threaded conversations have served as the foundation of virtual communities since the inception of the Internet. Usenet newsgroups, email lists, web boards, and discussion forums demonstrated the value of threaded conversation from the beginning.

**Prerequisite knowledge for Complete understanding and learning of Topic:  
( Max. Four important topics)**

- Basics of Internet
- Introduction of E-mail

**Detailed content of the Lecture:**

- More recent incarnations of threaded conversations show up in Facebook and LinkedIn group and profile page discussions, Reddit threads, GameSpot, Craigslist posts, YouTube comments, Amazon ratings, and Q&A sites like Quora and Stack Overflow.
- All contain collections of messages sent in reply to one another.
- The natural conversation style supported by the basic post-and-reply threaded message structure has proven enormously versatile, serving communities ranging widely in focus and goals.
- Cancer survivors and those seeking technical support or religious guidance are as likely to use a threaded discussion as a corporate workgroup.
- Although the basic structure of threaded conversation has remained surprisingly similar over time, conversations now include multimedia elements, user profiles (often with social network features), participation statistics, reputations scores, and ratings.
- Conversations are now attached to a host of other entities ranging from people (e.g., a public wall on a person's profile page) to items (e.g., movies; actors) to groups (e.g., university alumni) to events.
- This chapter primarily focuses on more traditional forums and email lists, but the core analysis techniques for threaded networks apply to other contexts.
- The threaded conversation structure lends itself well to network analysis, because a directed

link between individuals is created each time someone replies to another person's message.

- Unfortunately, most threaded conversation systems do not make this networked data easily accessible.
- The majority of threaded message content is not easily accessible because of the number of software platforms used and the fact that many groups only make content accessible to subscribed members.
- Many threaded message systems report participation statistics and ratings (e.g., top 10 contributors), which are important metrics. However, they fail to capture the social connections between members, a critical component of virtual communities and internal communities of practice.

**Video Content / Details of website for further learning (if any):**

- <https://en.wikipedia.org/wiki/Usenet>
- <https://www.sciencedirect.com/topics/computer-science/usenet-newsgroups>

**Important Books/Journals for further learning including the page nos.:**

**Net Reference**

**Course Faculty**

**Verified by HOD**



# MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)

Rasipuram - 637 408, Namakkal Dist., Tamil Nadu



LECTURE HANDOUTS

L 08

MCA

I / II

Course Code &Name : 19CAB11 & INTERNET AND JAVA PROGRAMMING

Name of the Faculty : Mrs. G. Krishnaveni

Year / Semester/Section : I / II

Date of Lecture: 03.03.2021

## Topic of Lecture: Voice and Video Conferencing

### Introduction : ( Maximum 5 sentences)

- Threaded conversations have served as the foundation of virtual communities since the inception of the Internet. Usenet newsgroups, email lists, web boards, and discussion forums demonstrated the value of threaded conversation from the beginning.

### Prerequisite knowledge for Complete understanding and learning of Topic: ( Max. Four important topics)

- Basics of Internet
- Introduction of E-mail

### Detailed content of the Lecture:

- Audio conferencing is where at least two individuals in various locations use technology like a conference bridge to hold an audio call.
- Audio conferencing is not quite the same as a traditional phone in that all participants dial into a central system that connects them rather than directly dialling each other.
- It aims at accomplishing communications and collaboration at the same time.
- [Many audio conferencing products](#) may accompany online collaboration elements (similar to screen-sharing capabilities), to additionally enhance the value of audio meetings.

### Video Conferencing

- When at least two individuals utilize digital platforms to communicate and collaborate with each other in order to accomplish a common goal adequately then it is known as a video conferencing.
- A unified and [simple video conferencing solution](#) which not just makes communication easy...
- Yet, it additionally lower the chances of having too many overlapping applications, simplifies troubleshooting and maintenance and saves everyone time and energy by lowering training needs.

### Benefits of Audio and Video Conferencing in Business:

#### 1. Reduces Travel Costs:

- It was not that long ago when all business meetings happened face-to-face, which involved travel, expense and time.
- However, through **audio and video conferencing** an organization can save a lot of time and money.

## **2. Keep Connected to Your Employees:**

- If you have employees working from home or out on the road, through audio video conference system you can stay in touch with them consistently.
- It's an extremely helpful way to keep in touch.

## **3. Increases Productivity:**

- If collaboration is done well – it can increase productivity essentially.
- **Audio and video conferencing** can be conducted at any time, so you don't need to waste additional time sorting out the meeting as you did in past.
- You can easily begin an audio and video call through your PC, mobile or another device essentially, increasing effectiveness and productivity.

## **4. Improves Teamwork:**

- If you have large teams or members of staff at various locations, video conferencing will assist to unite them.
- Employees can share data and collaborate to make a better-informed decision, which will prompt better working relationships internally.

## **5. Effective Communication:**

- Not just would you be able to hear people's voices, through video conferencing you can likewise see the people you are communicating to, see their expressions, instant responses and body language.
- By hopping on a quick virtual meeting, people can define tasks, goals and actions in detail so that every other person is on the same page.

## **6. Training Many People at a Time:**

- Organizations spend a lot of time and money on internal training programs.
- They tend to utilize traditional strategies for training their employees – such as a classrooms style session.
- However, a ton of time goes into arranging these sessions and guaranteeing there are sufficient resources for everyone.

## **Conclusion:**

- If you are thinking for an Audio and Video Conferencing Solution, you must ensure it can handle the demands of the current and remote workforce.

## **Video Content / Details of website for further learning (if any):**

- <https://vox-cpaas.com/blog/what-is-audio-and-video-conferencing/>
- <https://www.geeksforgeeks.org/audio-and-video-conferencing/>

## **Important Books/Journals for further learning including the page nos.:**

## **Net Reference**

Course Faculty

Verified by HOD





# MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)

Rasipuram - 637 408, Namakkal Dist., Tamil Nadu



LECTURE HANDOUTS

L 09

MCA

I / II

Course Code &Name : 19CAB11 & INTERNET AND JAVA PROGRAMMING

Name of the Faculty : Mrs. G. Krishnaveni

Year / Semester/Section : I / II

Date of Lecture: 04.03.2021

## Topic of Lecture: Web Security

### Introduction : ( Maximum 5 sentences)

- Cyber security is the protection of Internet-connected systems, including hardware, software, and data from cyber attackers.
- It is primarily about people, processes, and technologies working together to encompass the full range of threat reduction, vulnerability reduction, deterrence, international engagement, and recovery policies and activities, including computer network operations, information assurance, law enforcement, etc.

### Prerequisite knowledge for Complete understanding and learning of Topic: ( Max. Four important topics)

- Basics of Internet
- Introduction of E-mail

### Detailed content of the Lecture:

- It is the body of technologies, processes, and practices designed to protect networks, devices, programs, and data from attack, theft, damage, modification, or unauthorized access. Therefore, it may also be referred to as **information technology security**.
- Cyber-attack is now an international concern.
- It has given many concerns that could endanger the global economy. As the volume of cyber-attacks grows, companies and organizations, especially those that deal with information related to national security, health, or financial records, need to take steps to protect their sensitive business and personal information.
- This Cyber Security tutorial provides basic and advanced concepts of Cyber Security technology.
- It will cover the most popular concept of Cyber Security, such as what is Cyber Security, Cyber Security goals, types of cyber-attacks, types of cyber attackers, policies, digital signature, Cyber Security tools, security risk analysis, challenges, etc.
- Prerequisites
- It is a basic tutorial where we can quickly understand the topics discussed if we have a basic understanding of how a firm or organization handles computer security.
- It is also helpful for us to have some prior experience with computer updates, firewalls, antiviruses, and other security measures.
- Audience
- Our Cyber Security tutorial is designed to help beginners and professionals.
- Problems
- We assure you that you will not find any problem with this tutorial. However, if you find any, you can post it on the contact form.
- The technique of protecting internet-connected systems such as computers, servers, mobile



devices, electronic systems, networks, and data from malicious attacks is known as cyber security.

- We can divide cyber security into two parts one is cyber, and the other is security. Cyber refers to the technology that includes systems, networks, programs, and data.
- And security is concerned with the protection of systems, networks, applications, and information.
- In some cases, it is also called electronic information security or information technology security.
- Types of Cyber Security
- Every organization's assets are the combinations of a variety of different systems.
- These systems have a strong cyber security posture that requires coordinated efforts across all of its systems.
- Therefore, we can categorize cyber security in the following sub-domains:
- **Network Security:** It involves implementing the hardware and software to secure a computer network from unauthorized access, intruders, attacks, disruption, and misuse. This security helps an organization to protect its assets against external and internal threats.
- **Application Security:** It involves protecting the software and devices from unwanted threats. This protection can be done by constantly updating the apps to ensure they are secure from attacks. Successful security begins in the design stage, writing source code, validation, threat modeling, etc., before a program or device is deployed.
- **Information or Data Security:** It involves implementing a strong data storage mechanism to maintain the integrity and privacy of data, both in storage and in transit.
- **Identity management:** It deals with the procedure for determining the level of access that each individual has within an organization.
- **Operational Security:** It involves processing and making decisions on handling and securing data assets.
- **Mobile Security:** It involves securing the organizational and personal data stored on mobile devices such as cell phones, computers, tablets, and other similar devices against various malicious threats. These threats are unauthorized access, device loss or theft, malware, etc.
- **Cloud Security:** It involves in protecting the information stored in the digital environment or cloud architectures for the organization. It uses various cloud service providers such as AWS, Azure, Google, etc., to ensure security against multiple threats.
- **Disaster Recovery and Business Continuity Planning:** It deals with the processes, monitoring, alerts, and plans to how an organization responds when any malicious activity is causing the loss of operations or data. Its policies dictate resuming the lost operations after any disaster happens to the same operating capacity as before the event.
- **User Education:** It deals with the processes, monitoring, alerts, and plans to how an organization responds when any malicious activity is causing the loss of operations or data. Its policies dictate resuming the lost operations after any disaster happens to the same operating capacity as before the event.

**Video Content / Details of website for further learning (if any):**

- <https://www.w3schools.com/cybersecurity/index.php>
- <https://www.javatpoint.com/cyber-security-tutorial>

**Important Books/Journals for further learning including the page nos.:**

**Net Reference**

**Course Faculty**

**Verified by HOD**



# MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)  
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu



L - 10

## LECTURE HANDOUTS

MCA

I/II

Course Name with Code : 19CAB11 - INTERNET AND JAVA PROGRAMMING

Course Faculty : Mrs. G. Krishnaveni

Unit : JAVA FUNDAMENTALS

Date of Lecture: 05.03.2021

### Topic of Lecture: Java Features

#### Introduction : ( Maximum 5 sentences)

- The primary objective of [Java programming](#) language creation was to make it portable, simple and secure programming language.

#### Prerequisite knowledge for Complete understanding and learning of Topic:

- Basic of c/c++

#### Detailed content of the Lecture:

- **Object-oriented**
- Java is an **object-oriented** programming language.
- Everything in Java is an object.
- Object-oriented means we organize our software as a combination of different types of objects that incorporate both data and behavior.
- Object-oriented programming (OOPs) is a methodology that simplifies software development and maintenance by providing some rules.

Basic concepts of OOPs are:**Object**

- **Class**
- **Inheritance**
- **Polymorphism**
- **Abstraction**

#### Simple

- Java is very easy to learn, and its syntax is simple, clean and easy to understand.
- According to Sun Micro system, Java language is a simple programming language because: Java syntax is based on C++ (so easier for programmers to learn it after C++).
- Java has removed many complicated and rarely-used features, for example, explicit pointers, operator overloading, etc.

- There is no need to remove unreferenced objects because there is an Automatic Garbage Collection in Java.

### Platform Independent



### Secured

Java is best known for its security. With Java, we can develop virus-free systems. Java is secured because:

- **No explicit pointer**
- **Java Programs run inside a virtual machine sandbox**
- **Class loader:** Class loader in Java is a part of the Java Runtime Environment (JRE) which is used to load Java classes into the Java Virtual Machine dynamically. It adds security by separating the package for the classes of the local file system from those that are imported from network sources.
- **Bytecode Verifier:** It checks the code fragments for illegal code that can violate access rights to objects.
- **Security Manager:** It determines what resources a class can access such as reading and writing to the local disk.

### Robust

The English meaning of Robust is strong. Java is robust because:

- It uses strong memory management.
- There is a lack of pointers that avoids security problems.
- Java provides automatic garbage collection which runs on the Java Virtual Machine to get rid of objects which are not being used by a Java application anymore.
- There are exception handling and the type checking mechanism in Java. All these points make Java robust.

### Architecture-neutral

- Java is architecture neutral because there are no implementation dependent features, for example, the size of primitive types is fixed.
- In C programming, int data type occupies 2 bytes of memory for 32-bit architecture and 4 bytes of memory for 64-bit architecture. However, it occupies 4 bytes of memory for both 32 and 64-bit architectures in Java.

### Portable

- Java is portable because it facilitates you to carry the Java bytecode to any platform. It doesn't require any implementation.

### High-performance

- Java is faster than other traditional interpreted programming languages because Java bytecode is "close" to native code.
- It is still a little bit slower than a compiled language (e.g., C++). Java is an interpreted language that is why it is slower than compiled languages, e.g., C, C++, etc

### Distributed

- Java is distributed because it facilitates users to create distributed applications in Java.
- RMI and EJB are used for creating distributed applications.
- This feature of Java makes us able to access files by calling the methods from any machine on the internet.

### Multi-threaded

- A thread is like a separate program, executing concurrently.
- We can write Java programs that deal with many tasks at once by defining multiple threads.
- The main advantage of multi-threading is that it doesn't occupy memory for each thread.
- It shares a common memory area. Threads are important for multi-media, Web applications, etc.

### Dynamic

- Java is a dynamic language. It supports the dynamic loading of classes. It means classes are loaded on demand.
- It also supports functions from its native languages, i.e., C and C++.
- Java supports dynamic compilation and automatic memory management (garbage collection).

### Video Content / Details of website for further learning (if any):

- <https://www.javatpoint.com/features-of-java>
- <https://youtu.be/O5hShUO6wxs?t=12>

### Important Books/Journals for further learning including the page nos.:

- H.M. Deitel,P.J.Deitel, "Java How to Program" Prentice – Hall of India ,Newdelhi,  
Page No - 9

**Course Faculty**

**Verified by HOD**



# MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)  
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu



L - 11

## LECTURE HANDOUTS

MCA

I/II

**Course Name with Code : 19CAB11 - INTERNET AND JAVA PROGRAMMING**

**Course Faculty : Mrs.G.Krishnaveni**

**Unit : JAVA FUNDAMENTALS**

**Date of Lecture: 06.03.2021**

### Topic of Lecture: Java Platform

#### Introduction : ( Maximum 5 sentences)

- All Java platforms consist of a Java Virtual Machine (VM) and an application programming interface (API).

#### Prerequisite knowledge for Complete understanding and learning of Topic:

- What is java/Basics of java

#### Detailed content of the Lecture:

- There are four platforms of the Java programming language:
- Java Platform, Standard Edition (Java SE)
- Java Platform, Enterprise Edition (Java EE)
- Java Platform, Micro Edition (Java ME)
- Java FX
- The Java Virtual Machine is a program, for a particular hardware and software platform, that runs Java technology applications.
- An API is a collection of software components that you can use to create other software components or applications.
- Each Java platform provides a virtual machine and an API, and this allows applications written for that platform to run on any compatible system with all the advantages of the Java programming language: platform-independence, power, stability, ease-of-development, and security.

#### Java SE

- When most people think of the Java programming language, they think of the Java SE API.
- Java SE's API provides the core functionality of the Java programming language. It defines everything from the basic types and objects of the Java programming language to high-level classes that are used for networking, security, database access, graphical user interface (GUI) development, and XML parsing.
- In addition to the core API, the Java SE platform consists of a virtual machine, development tools, deployment technologies, and other class libraries and toolkits commonly used in Java

technology applications.

### **Java EE**

- The Java EE platform is built on top of the Java SE platform. The Java EE platform provides an API and runtime environment for developing and running large-scale, multi-tiered, scalable, reliable, and secure network applications.

### **Java ME**

- The Java ME platform provides an API and a small-footprint virtual machine for running Java programming language applications on small devices, like mobile phones.
- The API is a subset of the Java SE API, along with special class libraries useful for small device application development. Java ME applications are often clients of Java EE platform services.

### **Java FX**

- Java FX technology is a platform for creating rich internet applications written in Java FX Script™.
- Java FX Script is a statically-typed declarative language that is compiled to Java technology bytecode, which can then be run on a Java VM.
- Applications written for the Java FX platform can include and link to Java programming language classes, and may be clients of Java EE platform services.

### **Video Content / Details of website for further learning (if any):**

- <https://docs.oracle.com/cd/E19798-01/821-1770/gcrkk/index.html>
- <https://www.youtube.com/watch?v=uicd3mMEBf0>

### **Important Books/Journals for further learning including the page nos.:**

- H.M. Deitel,P.J.Deitel, ”Java How to Program” Prentice – Hall of India ,Newdelhi, Page No - 10
- 

**Course Faculty**

**Verified by HOD**



# MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)

Rasipuram - 637 408, Namakkal Dist., Tamil Nadu



L - 12

LECTURE HANDOUTS

MCA

I/II

Course Name with Code : 19CAB11 - INTERNET AND JAVA PROGRAMMING

Course Faculty : Mrs. G. Krishnaveni

Unit : JAVAFUNDAMENTALS

Date of Lecture: 08.03.2021

## Topic of Lecture: Java Fundamentals-Expression

### Introduction : ( Maximum 5 sentences)

- An *expression* is a construct made up of variables, operators, and method invocations, which are constructed according to the syntax of the language, that evaluates to a single value.

### Prerequisite knowledge for Complete understanding and learning of Topic:

- Basic of java

### Detailed content of the Lecture:

- An *expression* is a construct made up of variables, operators, and method invocations, which are constructed according to the syntax of the language, that evaluates to a single value. You've already seen examples of expressions, illustrated in bold below:
  - `int cadence = 0;`
  - `anArray[0] = 100;`
  - `System.out.println("Element 1 at index 0: " + anArray[0]);`
  - `int result = 1 + 2; // result is now 3`
  - `if (value1 == value2)`
  - `System.out.println("value1 == value2");`
- The data type of the value returned by an expression depends on the elements used in the expression. The expression `cadence = 0` returns an `int` because the assignment operator returns a value of the same data type as its left-hand operand; in this case, `cadence` is an `int`. As you can see from the other expressions, an expression can return other types of values as well, such as `boolean` or `String`.
- The Java programming language allows you to construct compound expressions from various smaller expressions as long as the data type required by one part of the expression matches the data type of the other. Here's an example of a compound expression:
  - `1 * 2 * 3`
  - In this particular example, the order in which the expression is evaluated is unimportant because the result of multiplication is independent of order; the outcome is always the same, no matter in which order you apply the multiplications. However, this is not true of all expressions. For example, the following expression gives different results, depending on whether you

perform the addition or the division operation first:

- `x + y / 100` // ambiguous
- You can specify exactly how an expression will be evaluated using balanced parenthesis: ( and ). For example, to make the previous expression unambiguous, you could write the following:

- `(x + y) / 100` // unambiguous, recommended
- If you don't explicitly indicate the order for the operations to be performed, the order is determined by the precedence assigned to the operators in use within the expression. Operators that have a higher precedence get evaluated first.
- For example, the division operator has a higher precedence than does the addition operator. Therefore, the following two statements are equivalent:

`x + y / 100`

- `x + (y / 100)` // unambiguous, recommended
- When writing compound expressions, be explicit and indicate with parentheses which operators should be evaluated first. This practice makes code easier to read and to maintain.

**Video Content / Details of website for further learning (if any):**

- <https://docs.oracle.com/javase/tutorial/java/nutsandbolts/expressions.html>

**Important Books/Journals for further learning including the page nos.:**

- H.M. Deitel,P.J.Deitel, ”Java How to Program” Prentice – Hall of India ,Newdelhi, Page No - 36

**Course Faculty**

**Verified by HOD**





# MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)  
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu



L - 13

## LECTURE HANDOUTS

MCA

I/II

Course Name with Code : 19CAB11 - INTERNET AND JAVA PROGRAMMING

Course Faculty : Mrs.G. Krishnaveni

Unit : JAVA FUNDAMENTALS

Date of Lecture: 09.03.2021

### Topic of Lecture: Java Fundamentals- Operators

#### Introduction : ( Maximum 5 sentences)

Java provides a rich operator environment. We can classify the basic operators in java in the following groups:

- Arithmetic Operators
- Relational Operators
- Bitwise Operators
- Assignment Operators
- Logical Operators

#### Prerequisite knowledge for Complete understanding and learning of Topic:

- Basic of java/maths

#### Detailed content of the Lecture:

**1. Arithmetic Operators:** Arithmetic operators are used to perform arithmetic/mathematical operations on operands.

- **Addition ('+')** : Adds two operands
- **Subtraction ('-')**: Subtracts two operands
- **Multiplication ('\*')**: Multiplies two operands
- **Division ('/')**: Divides the first operand by the second.
- **Modulus ('%')**: Returns the remainder when first operand is divided by the second
- **Increment ('++')**: Increment the value of an integer. When placed before the variable name (also called pre-increment operator), its value is incremented instantly. For example, ++x. And when it is placed after the variable name (also called post-increment operator), its value is preserved temporarily until the execution of this statement and it gets updated before the execution of the next statement. For example, x++.
- **Decrement ('--')**: Decrement the value of an integer. When placed before the variable name (also called pre-decrement operator), its value is decremented instantly.
- For example, --x. And when it is placed after the variable name (also called post-decrement operator), its value is preserved temporarily until the execution of this statement and it gets updated before the execution of the next statement. For example, x--.

//Java program to explain arithmetic operators

```
import java.util.*;
```

```
class A{
```

```

public static void main(String args[])
{

    int a = 10, b = 4, res;

    //printing a and b
    System.out.println("a is "+a+ " and b is "+ b);

    res = a+b; //addition
    System.out.println("a+b is "+res);

    res = a-b; //subtraction
    System.out.println("a-b is "+res);

    res = a*b; //multiplication
    System.out.println("a*b is "+res);

    res = a/b; //division
    System.out.println("a/b is "+res);

    res = a%b; //modulus
    System.out.println("a%b is "+res);

}
}

```

**Output:**

a is 10 and b is 4

a+b is 14

a-b is 6

a\*b is 40

a/b is 2

a%b is 2

**2. Relational Operators:** The relational operators determine the relationship that one operand has to the other. The relational operators evaluates the relation between two operations and returns *true* if the relation exists else *false*.

- **'==' operator:** checks whether the two given operands are equal or not. If so, it returns true. Otherwise it returns false. For example, **5==5** will return true.
- **'!=' operator:** checks whether the two given operands are equal or not. If not, it returns true. Otherwise it returns false. It is the exact boolean complement of the '==' operator. For example, **5!=5** will return false.
- **'>' operator:** checks whether the first operand is greater than the second operand. If so, it returns true. Otherwise it returns false. For example, **6>5** will return true.
- **'<' operator:** checks whether the first operand is lesser than the second operand. If so, it returns true. Otherwise it returns false. For example, **6<5** will return false.
- **'>=' operator:** checks whether the first operand is greater than or equal to the second operand. If so, it returns true. Otherwise it returns false. For example, **5>=5** will return true.
- **'<=' operator:** checks whether the first operand is lesser than or equal to the second operand. If so, it returns true. Otherwise it returns false. For example, **5<=5** will also return true.

**//Java program for relational operators**

```

import java.util.*;
class A{

    public static void main(String args[])
    {

        int a=10, b=4;

        // relational operators
        // greater than example
        if (a > b)
            System.out.println("a is greater than b");
        else System.out.println("a is less than or equal to b");

        // greater than equal to
        if (a >= b)
            System.out.println("a is greater than or equal to b");
        else System.out.println("a is lesser than b");

        // less than example
        if (a < b)
            System.out.println("a is less than b");
        else System.out.println("a is greater than or equal to b");

        // lesser than equal to
        if (a <= b)
            System.out.println("a is lesser than or equal to b");
        else System.out.println("a is greater than b");

        // equal to
        if (a == b)
            System.out.println("a is equal to b");
        else System.out.println("a and b are not equal");

        // not equal to
        if (a != b)
            System.out.println("a is not equal to b");
        else System.out.println("a is equal b");
    }
}

```

**Output:**

a is greater than b

a is greater than or equal to b

a is greater than or equal to b

a is greater than b

a and b are not equal

a is not equal to b

**3. Bitwise Operators:** Java provides several bitwise operators to work with integer types, **long**, **int**, **short**, **char**, **byte**. Bitwise operators performs bit-by-bit operation on binary

representation of integers. These operators act upon the individual bits of their operands.

For Example:

Assume  $a = 9$  and  $b = 7$ .

In binary form,

$a = 1001$

$b = 0111$

-----

$a \& b = 0001$

**4. Assignment Operator:** The assignment operator is used to assign value to a variable. The general form of an assignment operator is:

**var = expression**

Different ways of using assignment operator:

- ‘=’: This is the simplest assignment operator. It assigns the value of left operand to the right operand. For example,  $a = 3$ .
- ‘+=’: This operator first adds the left and right operands and then assigns the result to the left operand. For example,  $a += b$  is equivalent to  $a = a + b$ .
- ‘-=’: This operator first subtracts the right operand from left operand and then assign the result to left operand. For example,  $a -= b$  is equivalent to  $a = a - b$ .
- ‘\*=’: This operator first multiplies the right operand and left operand and then assign the result to left operand. For example,  $a *= b$  is equivalent to  $a = a * b$
- ‘/=’: This operator first divides the left operand by right operand and then assign the result to left operand. For example,  $a /= b$  is equivalent to  $a = a / b$
- ‘%=’: This operator calculates modulus using left and right operand and then assigns the result to the left operand. For example,  $a %= b$  is equivalent to  $a = a \% b$

Similarly, we can also use operators like , ^= , &= , |=.

**5. Logical Operators:** Logical operators perform logical operations like logical AND, logical OR etc. Let us assume that variable **a** holds the boolean value *true* and **b** holds the boolean value *false*.

Below are some logical operators we can use:

- **Logical AND (&&):** This operator will return true if both the left and right operands are true, otherwise it will return false. For example, **a && b** is *false*.
- **Logical OR (||):** This operator will return true if any one of the left and right operands are true. It will return false when both left and right operands are false. For example, **a || b** is *True*.
- **Logical NOT (!):** This is a unary operator and can be used with a single operand. This will return true if the operand is false and return false if the operand is true. For example, **!a** is *false* and **!b** is *true*.

**Video Content / Details of website for further learning (if any):**

- <https://www.geeksforgeeks.org/basic-operators-java>

**Important Books/Journals for further learning including the page nos.:**

- H.M. Deitel,P.J.Deitel, ”Java How to Program” Prentice – Hall of India ,Newdelhi, Page No - 202

**Course Faculty**

**Verified by HOD**



# MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)  
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu



LECTURE HANDOUTS

L - 14

MCA

I/II

Course Name with Code : 19CAB11 - INTERNET AND JAVA PROGRAMMING

Course Faculty : Mrs.G.Krishnaveni

Unit : JAVA FUNDAMENTALS

Date of Lecture: 10.03.2021

**Topic of Lecture:** Java Fundamentals- Control Structures

**Introduction : ( Maximum 5 sentences)**

- **Java** provides statements that can be used to control the flow of Java code. Such statements are called control flow statements. It is one of the fundamental features of Java, which provides a smooth flow of program.

**Prerequisite knowledge for Complete understanding and learning of Topic:**

- Basic of java/expression/operators

**Detailed content of the Lecture:**

Java provides three types of control flow statements.

1. Decision Making statements
  1. if statements
  2. switch statement
2. Loop statements
  1. do while loop
  2. while loop
  3. for loop
  4. for-each loop
3. Jump statements
  1. break statement
  2. continue statement

**Decision-Making statements:**

1) If Statement:

In Java, the "if" statement is used to evaluate a condition.

The control of the program is diverted depending upon the specific condition.

The condition of the If statement gives a Boolean value, either true or false. In Java, there are four types of if-statements given below

- Simple if statement
- if-else statement

- if-else-if ladder
- Nested if-statement

### 1) Simple if statement:

```
public class Student {
    public static void main(String[] args) {
        int x = 10;
        int y = 12;
        if(x+y > 20) {
            System.out.println("x + y is greater than 20");
        }
    }
}
```

#### Output:

```
x + y is greater than 20
```

### if-else statement

```
public class Student {
    public static void main(String[] args) {
        int x = 10;
        int y = 12;
        if(x+y < 10) {
            System.out.println("x + y is less than 10");
        } else {
            System.out.println("x + y is greater than 20");
        }
    }
}
```

#### Output:

```
x + y is greater than 20
```

### if-else-if ladder:

```
public class Student {
    public static void main(String[] args) {
        String city = "Delhi";
        if(city == "Meerut") {
            System.out.println("city is meerut");
        } else if (city == "Noida") {
            System.out.println("city is noida");
        } else if(city == "Agra") {
            System.out.println("city is agra");
        } else {
            System.out.println(city);
        }
    }
}
```

```
}  
}  
}
```

### Output:

```
Delhi
```

### Nested if-statement

```
public class Student {  
  public static void main(String[] args) {  
    String address = "Delhi, India";  
  
    if(address.endsWith("India")) {  
      if(address.contains("Meerut")) {  
        System.out.println("Your city is Meerut");  
      } else if(address.contains("Noida")) {  
        System.out.println("Your city is Noida");  
      } else {  
        System.out.println(address.split(",")[0]);  
      }  
    } else {  
      System.out.println("You are not living in India");  
    }  
  }  
}
```

### Output:

```
Delhi
```

### Switch Statement:

In Java, [Switch statements](#) are similar to if-else-if statements.

```
public class Student implements Cloneable {  
  public static void main(String[] args) {  
    int num = 2;  
    switch (num){  
      case 0:  
        System.out.println("number is 0");  
        break;  
      case 1:  
        System.out.println("number is 1");  
        break;  
      default:  
        System.out.println(num);
```

```
}  
}  
}
```

**Output:**

```
2
```

### Loop Statements

#### Java for loop

```
public class Calculation {  
  public static void main(String[] args) {  
    // TODO Auto-generated method stub  
    int sum = 0;  
    for(int j = 1; j<=10; j++) {  
      sum = sum + j;  
    }  
    System.out.println("The sum of first 10 natural numbers is " + sum);  
  }  
}
```

**Output:**

```
The sum of first 10 natural numbers is 55
```

#### Java for-each loop

```
public class Calculation {  
  public static void main(String[] args) {  
    // TODO Auto-generated method stub  
    String[] names = {"Java","C","C++","Python","JavaScript"};  
    System.out.println("Printing the content of the array names:\n");  
    for(String name:names) {  
      System.out.println(name);  
    }  
  }  
}
```

**Output:**

```
Printing the content of the array names:
```

```
Java
```

```
C
```

```
C++
```

```
Python
```

```
JavaScript
```

#### Java while loop



```
public class Calculation {
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        int i = 0;
        System.out.println("Printing the list of first 10 even numbers \n");
        while(i<=10) {
            System.out.println(i);
            i = i + 2;
        }
    }
}
```

**Output:**

```
Printing the list of first 10 even numbers
```

```
0
2
4
6
8
10
```

Java do-while loop

```
public class Calculation {
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        int i = 0;
        System.out.println("Printing the list of first 10 even numbers \n");
        do {
            System.out.println(i);
            i = i + 2;
        } while(i<=10);
    }
}
```

**Output:**

```
Printing the list of first 10 even numbers
```

```
0
2
4
```

6  
8  
10

**Video Content / Details of website for further learning (if any):**

- <https://www.javatpoint.com/control-flow-in-java>

**Important Books/Journals for further learning including the page nos.:**

- H.M. Deitel,P.J.Deitel, ”Java How to Program” Prentice – Hall of India ,Newdelhi,  
Page No - 180
- 

**Course Faculty**

**Verified by HOD**



# MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)  
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu



L - 15

## LECTURE HANDOUTS

MCA

I/II

Course Name with Code : 19CAB11 - INTERNET AND JAVA PROGRAMMING

Course Faculty : Mrs.G.Krishnaveni

Unit : JAVA FUNDAMENTALS

Date of Lecture: 11.03.2021

### Topic of Lecture: Java Fundamentals- Classes

#### Introduction : ( Maximum 5 sentences)

- A class is a blueprint from which individual objects are created

#### Prerequisite knowledge for Complete understanding and learning of Topic:

- Basic of OOPS/java

#### Detailed content of the Lecture:

- A class can contain any of the following variable types.
- **Local variables** – Variables defined inside methods, constructors or blocks are called local variables. The variable will be declared and initialized within the method and the variable will be destroyed when the method has completed.
- **Instance variables** – Instance variables are variables within a class but outside any method. These variables are initialized when the class is instantiated. Instance variables can be accessed from inside any method, constructor or blocks of that particular class.
- **Class variables** – Class variables are variables declared within a class, outside any method, with the static keyword.
- A class can have any number of methods to access the value of various kinds of methods. In the above example, barking(), hungry() and sleeping() are methods.
- Following are some of the important topics that need to be discussed when looking into classes of the Java Language.

#### Example

```

public class Dog {
    String breed;
    int age;
    String color;

```

```

    void barking() {
    }

```

```

    void hungry() {

```

- }

- `void sleeping(){`
- `}`
- `}`

**Video Content / Details of website for further learning (if any):**

- [https://www.tutorialspoint.com/java/java\\_object\\_classes.htm](https://www.tutorialspoint.com/java/java_object_classes.htm)
- 

**Important Books/Journals for further learning including the page nos.:**

- H.M. Deitel,P.J.Deitel, ”Java How to Program” Prentice – Hall of India ,Newdelhi,  
Page No - 82
- 

**Course Faculty**

**Verified by HOD**



# MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)  
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu



L - 16

## LECTURE HANDOUTS

MCA

I/II

Course Name with Code : 19CAB11 - INTERNET AND JAVA PROGRAMMING

Course Faculty : Mrs.G.Krishnaveni

Unit : JAVAFUNDAMENTALS

Date of Lecture: 12.03.2021

### Topic of Lecture: Packages and Interfaces

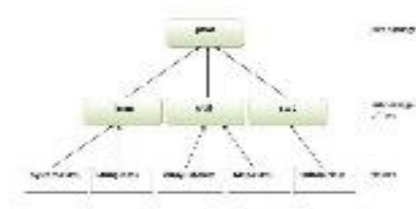
#### Introduction : ( Maximum 5 sentences)

- A **java package** is a group of similar types of classes, interfaces and sub-packages.
- Package in java can be categorized in two form, built-in package and user-defined package.
- There are many built-in packages such as java, lang, awt, javax, swing, net, io, util, sql etc.

#### Prerequisite knowledge for Complete understanding and learning of Topic:

- Java classes/import modules

#### Detailed content of the Lecture:



There are many built-in packages such as java, lang, awt, javax, swing, net, io, util, sql etc.

Here, we will have the detailed learning of creating and using user-defined packages.

#### Advantage of Java Package

- 1) Java package is used to categorize the classes and interfaces so that they can be easily maintained.
- 2) Java package provides access protection.
- 3) Java package removes naming collision.

#### Simple example of java package

The **package keyword** is used to create a package in java.

```
//save as Simple.java
```

```
package mypack;
public class Simple{
    public static void main(String args[]){
        System.out.println("Welcome to package");
    }
}
```

**To Compile:** javac -d . Simple.java

**To Run:** java mypack.Simple

**Output:** Welcome to package

There are three ways to access the package from outside the package.

1. import package.\*;
2. import package.classname;
3. fully qualified name.

### ***1) Using packagename.\****

If you use package.\* then all the classes and interfaces of this package will be accessible but not subpackages.

The import keyword is used to make the classes and interface of another package accessible to the current package.

### ***2) Using packagename.classname***

If you import package.classname then only declared class of this package will be accessible.

### ***3) Using fully qualified name***

If you use fully qualified name then only declared class of this package will be accessible. Now there is no need to import. But you need to use fully qualified name every time when you are accessing the class or interface.

***Sequence of the program must be package then import then class.***



## Subpackage in java

Package inside the package is called the **subpackage**. It should be created **to categorize the package further**

## Example of Subpackage

```
package com.javatpoint.core;
class Simple{
    public static void main(String args[]){
        System.out.println("Hello subpackage");
    }
}
```

**To Compile:** javac -d . Simple.java

**To Run:** java com.javatpoint.core.Simple

Output:Hello subpackage

**Video Content / Details of website for further learning (if any):**

- <https://www.javatpoint.com/package>

**Important Books/Journals for further learning including the page nos.:**

- H.M. Deitel,P.J.Deitel,"Java How to Program" Prentice – Hall of India ,Newdelhi,  
Page No - 243

**Course Faculty**

**Verified by HOD**



# MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)  
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu



L - 17

MCA

LECTURE HANDOUTS

I/II

Course Name with Code : 19CAB11 - INTERNET AND JAVA PROGRAMMING

Course Faculty : Mrs. G.Krishnaveni

Unit : JAVA FUNDAMENTALS

Date of Lecture: 13.03.2021

## Topic of Lecture: Packages and Interfaces

### Introduction : ( Maximum 5 sentences)

- An **interface in Java** is a blueprint of a class.
- It has static constants and abstract methods.
- The interface in Java is *a mechanism to achieve abstraction*.
- There can be only abstract methods in the Java interface, not method body. It is used to achieve abstraction and multiple inheritance in Java.

### Prerequisite knowledge for Complete understanding and learning of Topic:

- Oops concepts

### Detailed content of the Lecture:

- The interface in Java is *a mechanism to achieve abstraction*. There can be only abstract methods in the Java interface, not method body.
- It is used to achieve abstraction and multiple inheritance in Java.
- In other words, you can say that interfaces can have abstract methods and variables. It cannot have a method body.
- Java Interface also **represents the IS-A relationship**.
- It cannot be instantiated just like the abstract class.
- Since Java 8, we can have **default and static methods** in an interface.
- Since Java 9, we can have **private methods** in an interface.

### Why use Java interface?

There are mainly three reasons to use interface. They are given below.

It is used to achieve abstraction.

- By interface, we can support the functionality of multiple inheritance.
- It can be used to achieve loose coupling.

### How to declare an interface?



- An interface is declared by using the interface keyword.
- It provides total abstraction; means all the methods in an interface are declared with the empty body, and all the fields are public, static and final by default.
- A class that implements an interface must implement all the methods declared in the interface.

### Syntax:

```
interface <interface_name>{

    // declare constant fields

    // declare methods that abstract

    // by default.

}
```

**The relationship between classes and interfaces** As shown in the figure given below, a class extends another class, an interface extends another interface, but a class implements an interface.



```
interface printable{
void print();
}

class A6 implements printable{
public void print(){System.out.println("Hello");}

public static void main(String args[]){
A6 obj = new A6();
obj.print();
}
} Output:
```

Hello

### Video Content / Details of website for further learning (if any):

- <https://www.javatpoint.com/interface-in-java>

### Important Books/Journals for further learning including the page nos.:

- H.M. Deitel,P.J.Deitel,,"Java How to Program" Prentice – Hall of India ,Newdelhi, Page No - 245

Course Faculty

Verified by HOD



# MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)  
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu



L - 18

LECTURE HANDOUTS

MCA

I/II

Course Name with Code : 19CAB11 - INTERNET AND JAVA PROGRAMMING

Course Faculty : Mrs.G.Krishnaveni

Unit : JAVA FUNDAMENTALS

Date of Lecture: 15.03.2021

## Topic of Lecture: Exception Handling

### Introduction : ( Maximum 5 sentences)

- The **Exception Handling in Java** is one of the powerful *mechanism to handle the runtime errors* so that the normal flow of the application can be maintained.

### Prerequisite knowledge for Complete understanding and learning of Topic:

- Basic of java/syntax

### Detailed content of the Lecture:

- Exception Handling is a mechanism to handle runtime errors such as `ClassNotFoundException`, `IOException`, `SQLException`, `RemoteException`, etc.

### Advantage of Exception Handling

The core advantage of exception handling is to **maintain the normal flow of the application**. An exception normally disrupts the normal flow of the application; that is why we need to handle exceptions. Let's consider a scenario:

- statement 1;
- statement 2;
- statement 3;
- statement 4;
- statement 5;*//exception occurs*
- statement 6;
- statement 7;
- statement 8;
- statement 9;
- statement 10;

Suppose there are 10 statements in a Java program and an exception occurs at statement 5; the rest of the code will not be executed, i.e., statements 6 to 10 will not be executed. However, when we perform exception handling, the rest of the statements will be executed. That is why

we use exception handling in [Java](#).

## Types of Java Exceptions

There are mainly two types of exceptions: checked and unchecked. An error is considered as the unchecked exception. However, according to Oracle, there are three types of exceptions namely:

1. Checked Exception
2. Unchecked Exception
3. Error

## Difference between Checked and Unchecked Exceptions

### 1) Checked Exception

The classes that directly inherit the Throwable class except RuntimeException and Error are known as checked exceptions. For example, IOException, SQLException, etc. Checked exceptions are checked at compile-time.

### 2) Unchecked Exception

The classes that inherit the RuntimeException are known as unchecked exceptions. For example, ArithmeticException, NullPointerException, ArrayIndexOutOfBoundsException, etc. Unchecked exceptions are not checked at compile-time, but they are checked at runtime.

### 3) Error

Error is irrecoverable. Some example of errors are OutOfMemoryError, VirtualMachineError, AssertionError etc.

## Java Exception Keywords

Java provides five keywords that are used to handle the exception. The following table describes each.

Keyword	Description
try	The "try" keyword is used to specify a block where we should place an exception code. we can't use try block alone. The try block must be followed by either catch or finally.
catch	The "catch" block is used to handle the exception. It must be preceded by try block which we can't use catch block alone. It can be followed by finally block later.
finally	The "finally" block is used to execute the necessary code of the program. It is executed whether exception is handled or not.
throw	The "throw" keyword is used to throw an exception.
throws	The "throws" keyword is used to declare exceptions. It specifies that there may occur an exception in the method. It doesn't throw an exception. It is always used with method signature.

## JavaExceptionExample.java

```
public class JavaExceptionExample{
    public static void main(String args[]){
        try{
            //code that may raise exception
            int data=100/0;
        }catch(ArithmeticException e){System.out.println(e);}
        //rest code of the program
        System.out.println("rest of the code...");
    }
}
```

### Output:

```
Exception in thread main java.lang.ArithmeticException:/ by zero
rest of the code...
```

### Video Content / Details of website for further learning (if any):

- <https://www.javatpoint.com/exception-handling-in-java>

### Important Books/Journals for further learning including the page nos.:

- H.M. Deitel,P.J.Deitel,"Java How to Program" Prentice – Hall of India ,Newdelhi, Page No - 638

Course Faculty

Verified by HOD



# MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)  
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu



L - 19

MCA

LECTURE HANDOUTS

I/II

Course Name with Code : 19CAB11 - INTERNET AND JAVA PROGRAMMING

Course Faculty : Mrs. G. Krishnaveni

Unit : PACKAGES

Date of Lecture: 16.03.2021

## Topic of Lecture: AWT Package

### Introduction : ( Maximum 5 sentences)

- Java AWT (*Abstract Window Toolkit*) is an API to develop Graphical User Interface (GUI) or windows-based applications in Java.

### Prerequisite knowledge for Complete understanding and learning of Topic:

- Java and packages

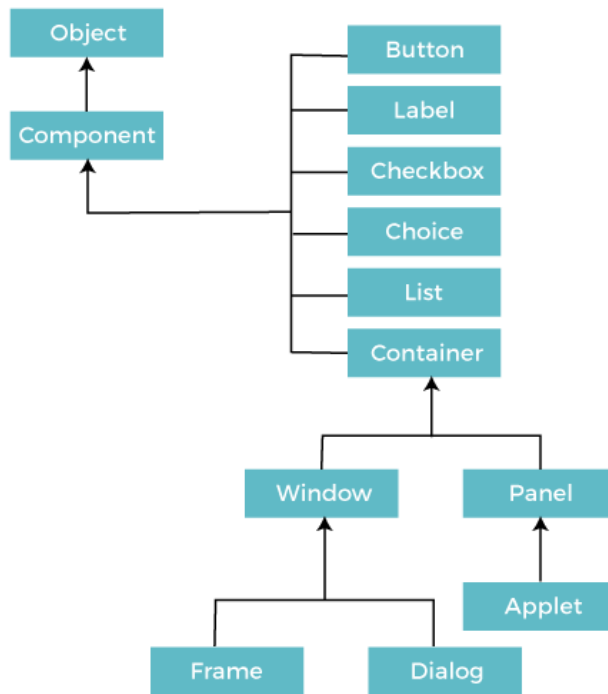
### Detailed content of the Lecture:

- Java AWT components are platform-dependent i.e. components are displayed according to the view of operating system.
- AWT is heavy weight i.e. its components are using the resources of underlying operating system (OS).
- The java.awt package provides classes for AWT API such as TextField, Label, TextArea, RadioButton, CheckBox, Choice, List etc.

### Why AWT is platform independent?

- Java AWT calls the native platform calls the native platform (operating systems) subroutine for creating API components like TextField, ChechBox, button, etc.
- For example, an AWT GUI with components like TextField, label and button will have different look and feel for the different platforms like Windows, MAC OS, and Unix.
- The reason for this is the platforms have different view for their native components and AWT directly calls the native subroutine that creates those components.
- In simple words, an AWT application will look like a windows application in Windows OS whereas it will look like a Mac application in the MAC OS.

Hierarchy of Java AWT classes are given below.



## Components

All the elements like the button, text fields, scroll bars, etc. are called components. In Java AWT, there are classes for each component as shown in above diagram. In order to place every component in a particular position on a screen, we need to add them to a container.

## Container

The Container is a component in AWT that can contain another components like [buttons](#), textfields, labels etc. The classes that extends Container class are known as container such as **Frame**, **Dialog** and **Panel**. It is basically a screen where the where the components are placed at their specific locations. Thus it contains and controls the layout of components.

There are four types of containers in Java AWT:

1. Window
2. Panel
3. Frame
4. Dialog

## Window

The window is the container that have no borders and menu bars. You must use frame, dialog or another window for creating a window. We need to create an instance of Window class to create this container.

## Panel

The Panel is the container that doesn't contain title bar, border or menu bar. It is generic container for holding the components. It can have other components like button, text field etc. An instance of Panel class creates a container, in which we can add components.

## Frame

The Frame is the container that contain title bar and border and can have menu bars. It can have other components like button, text field, scrollbar etc. Frame is most widely used container while developing an AWT application.

### Useful Methods of Component Class

Method	Description
public void add(Component c)	Inserts a component on this component.
public void setSize(intwidth,int height)	Sets the size (width and height) of the component.
public void setLayout(LayoutManager m)	Defines the layout manager for the component.
public void setVisible(boolean status)	Changes the visibility of the component, by default false

## Java AWT

To create simple AWT example, you need a frame. There are two ways to create a GUI using Frame in AWT. By extending Frame class (**inheritance**)

1. By creating the object of Frame class (**association**)

### Video Content / Details of website for further learning (if any):

- <https://www.javatpoint.com/java-awt>
- <https://www.youtube.com/watch?v=IPoUcWhapLY>
- <https://www.youtube.com/watch?v=BdgGefwaEvo>

### Important Books/Journals for further learning including the page nos.:

- H.M. Deitel,P.J.Deitel,"Java How to Program" Prentice – Hall of India ,Newdelhi, Page No -515

Course Faculty

Verified by HOD



# MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)  
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu



L - 20

LECTURE HANDOUTS

MCA

I/II

Course Name with Code : 19CAB11 - INTERNET AND JAVA PROGRAMMING

Course Faculty : Mrs.G.Krishnaveni

Unit : PACKAGES

Date of Lecture: 17.03.2021

## Topic of Lecture: Layouts

### Introduction : ( Maximum 5 sentences)

- Layouts allow you to format components on the screen in a platform-independent way. Without layouts, you would be forced to place components at explicit locations on the screen, creating obvious problems for programs that need to run on multiple platforms.

### Prerequisite knowledge for Complete understanding and learning of Topic:

- Java packages/AWT package

### Detailed content of the Lecture:

- Layout managers try to give programs a consistent and reasonable appearance, regardless of the platform, the screen size, or actions the user might take.
- Every container has a `LayoutManager` that is responsible for positioning the component objects within it, regardless of the platform or the screen size.
- Layout managers eliminate the need to compute component placement on your own, which would be a losing proposition since the size required for any component depends on the platform on which it is displayed.
- The standard JDK provides 5 classes that implement the `LayoutManager` interface.
- They are `FlowLayout`, `GridLayout`, `BorderLayout`, `CardLayout`, and `GridBagLayout`.

### 1. FlowLayout

- The `FlowLayout` is the default layout for the `Panel` class, that includes its most famous subclass, `Applet`.
- When there are too many components to put, they "wrap" to a new row, similar to a word processor with word wrap enabled.
- When you add components to the screen, they move left to right (centred within the applet) based upon the order added and the width of the applet.
- If you resize an applet then the component's move will change based upon the new width and height. The following shows an example of both before and after resizing.

```
1. import java.awt.*;
2. import java.awt.event.*;
3. public class FLExample
4. {
```

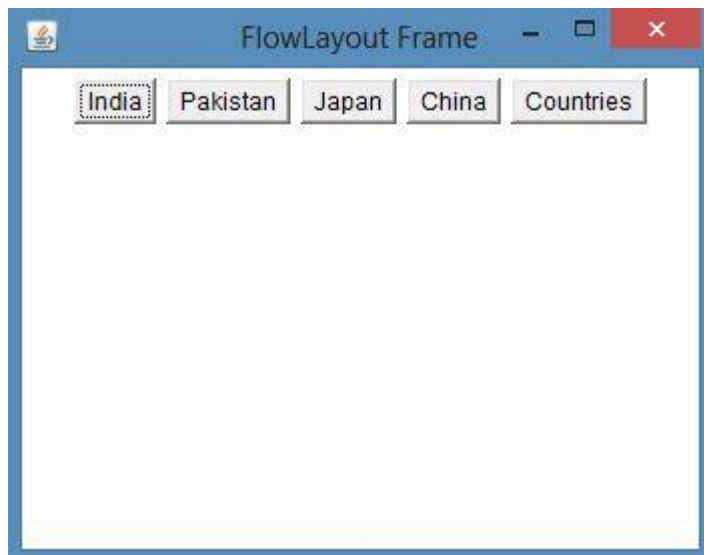


```

5. public static void main(String[] args)
6. {
7.     Frame frame= new Frame("FlowLayout Frame");
8.     Panel pa= new Panel();
9.     Button ba1= new Button();
10.    Button ba2=new Button();
11.    Button ba3=new Button();
12.    Button ba4=new Button();
13.    Button ba5=new Button();
14.    frame.add(pa);
15.    pa.setLayout(new FlowLayout());
16.    pa.add(new Button("India"));
17.    pa.add(new Button("Pakistan"));
18.    pa.add(new Button("Japan"));
19.    pa.add(new Button("China"));
20.    pa.add(new Button("Countries"));
21.    frame.setSize(300,300);
22.    frame.setVisible(true);
23.    frame.addWindowListener(new WindowAdapter()
24.    {
25.        public void windowClosing(WindowEvent e)
26.        {
27.            System.exit(0);
28.        }
29.    });
30. }
31. }

```

## Output



## 2. GridLayout

You start at row one, column one, then move across the row until it's full, then continue on to the next row. The GridLayout is widely used for arranging components in rows and columns. GridLayout can reposition or resize objects after adding or removing components

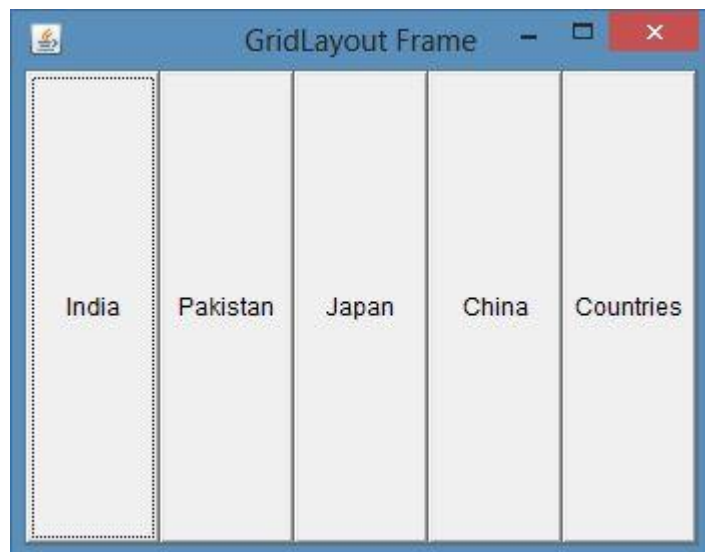
The same as FlowLayout, some changes exist in Line 3, Line 7 & Line 15.

**Line 3:** public class GLExample

**Line 7:** Frame frame= new Frame("GridLayout Frame");

**Line 15:** pa.setLayout(new GridLayout());

**Output**



### 3. BorderLayout

When you add a component to the layout, you must specify which area to place it in. BorderLayout is one of the more unusual layouts provided. It is the default layout for Window, along with its children, Frame and Dialog. BorderLayout provides 5 areas to hold components. These areas are named after the four different borders of the screen, North, South, East, and West, with any remaining space going into the Center area.

The same as FlowLayout, some changes exist in Line 3, Line 7 & Line 15-21.

**Line 3-**public class BLExample

**Line 7-** Frame frame= new Frame("BorderLayout Frame");

**Line 15-21**

```
pa.setLayout(new BorderLayout());
```

```
pa.add(new Button("India"), BorderLayout.NORTH);
```

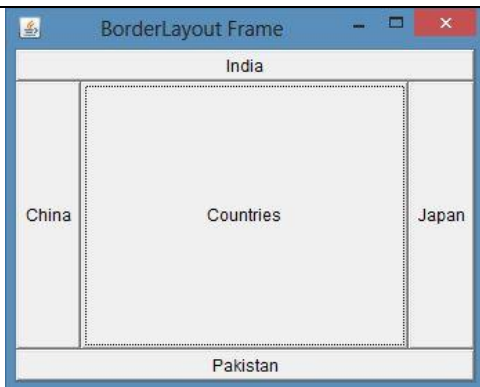
```
pa.add(new Button("Pakistan"), BorderLayout.SOUTH);
```

```
pa.add(new Button("Japan"), BorderLayout.EAST);
```

```
pa.add(new Button("China"), BorderLayout.WEST);
```

```
pa.add(new Button("Countries"), BorderLayout.CENTER);
```

**Output**



#### 4. CardLayout

A CardLayout usually manages several components, displaying one of them at a time and hiding the rest. CardLayout lets you assign names to the components it manages and lets you jump to a component by name. With a little work, you can use the CardLayout to create tabbed dialogue boxes or property sheets, that are not currently part of AWT. The CardLayout is a bit on the strange side. All the components are given the same size.

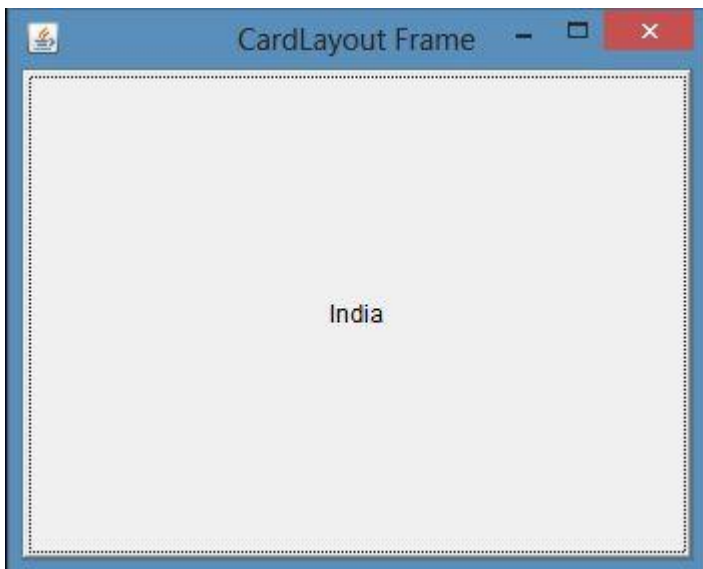
The same as FlowLayout, some changes exist in Line 3, Line 7 and Line 15:

**Line 3**-public class CLExample

**Line 7**: Frame frame= new Frame("CardLayout Frame");

**Line 15**: pa.setLayout(new CardLayout());

**Output:**



#### 5. GridBagLayout

You provide all the details of each component through instances of the GridBagConstraints class. GridBagLayout is the most sophisticated and complex of the layouts provided in the development kit. The same as FlowLayout, some changes exist in Line3, Line 7 & Line 15.

**Line 3**-public class CBLExample

**Line 7**- Frame frame= new Frame("GridBagLayout Frame");

**Line 15**- pa.setLayout(new GridBagLayout());

**Output**



**Video Content / Details of website for further learning (if any):**

- <https://www.c-sharpcorner.com/UploadFile/fd0172/layout-in-awt/>
- <https://www.youtube.com/watch?v=LpoXIyakmrQ>

**Important Books/Journals for further learning including the page nos.:**

- H.M. Deitel,P.J.Deitel, ”Java How to Program” Prentice – Hall of India ,Newdelhi,  
Page No - 567

**Course Faculty**

**Verified by HOD**



# MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)  
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu



LECTURE HANDOUTS

L - 21

MCA

I/II

Course Name with Code :19CAB11 - INTERNET AND JAVA PROGRAMMING

Course Faculty : Mrs.G.Krishnaveni

Unit : PACKAGES

Date of Lecture: 18.03.2021

## Topic of Lecture: Containers

### Introduction : ( Maximum 5 sentences)

- The runtime environment for JavaEE (j2ee) applications. The client/user can request only a static WebPages from the server.
- If the user wants to read the web pages as per input then the servlet container is used in java.

### Prerequisite knowledge for Complete understanding and learning of Topic:

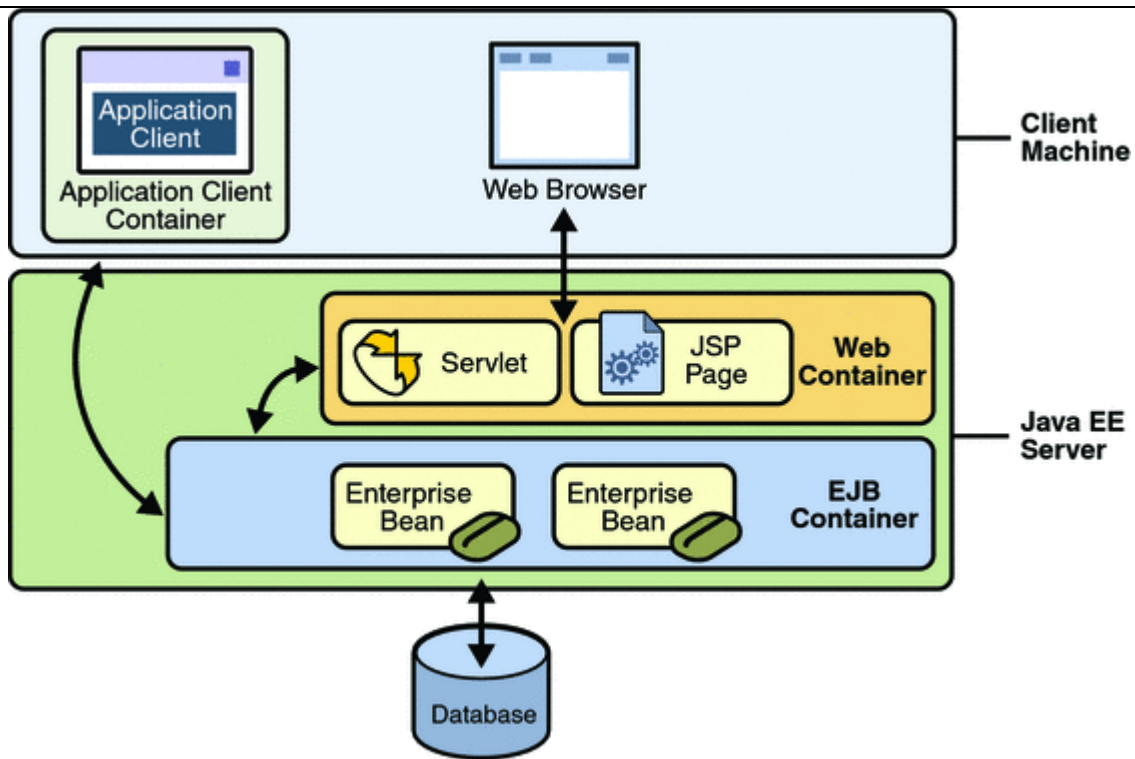
- Java packages/AWT/Layouts

### Detailed content of the Lecture:

- "Containers are **the interface between a component and the low-level platform-specific functionality that supports the component.**

### Container Types

- Java EE server: The runtime portion of a Java EE product. ...
- Enterprise JavaBeans (EJB) container: Manages the execution of enterprise beans for Java EE applications. ...
- Web container: Manages the execution of JSP page and servlet components for Java EE applications.
- **Application client container:** Manages the execution of application client components. Application clients and their container run on the client.
- **Applet container:** Manages the execution of applets. Consists of a web browser and Java Plug-in running on the client together.



### Servlet Container States

- The servlet container is the part of web server which can be run in a separate process. We can classify the servlet container states in three types:
- **Standalone:** It is typical Java-based servers in which the servlet container and the web servers are the integral part of a single program. For example:- Tomcat running by itself
- **In-process:** It is separated from the web server, because a different program runs within the address space of the main server as a plug-in. For example:- Tomcat running inside the JBoss.
- **Out-of-process:** The web server and servlet container are different programs which are run in a different process. For performing the communications between them, web server uses the plug-in provided by the servlet container.

### The Servlet Container performs many operations that are given below:

- Life Cycle Management
- Multithreaded support
- Object Pooling
- Security etc.

**Video Content / Details of website for further learning (if any):**

- <https://www.javatpoint.com/container>
- <https://www.youtube.com/watch?v=usqQLn21ShU>

**Important Books/Journals for further learning including the page nos.:**

- H.M. Deitel,P.J.Deitel,"Java How to Program" Prentice – Hall of India ,Newdelhi,  
Page No - 536

**Course Faculty**

**Verified by HOD**



# MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)

Rasipuram - 637 408, Namakkal Dist., Tamil Nadu



## LECTURE HANDOUTS

MCA

I/II

L - 22

Course Name with Code : 19CAB11 - INTERNET AND JAVA PROGRAMMING

Course Faculty : Mrs. G.Krishnaveni

Unit : PACKAGES

Date of Lecture: 19.03.2021

### Topic of Lecture: Containers

#### Introduction : ( Maximum 5 sentences)

- Server is a device or a computer program that accepts and responds to the request made by other program, known as client.
- It is used to manage the network resources and for running the program or software that provides services.

#### Prerequisite knowledge for Complete understanding and learning of Topic:

- Container and server AWT Web /server

#### Detailed content of the Lecture:

Server is a device or a computer program that accepts and responds to the request made by other program, known as client. It is used to manage the network resources and for running the program or software that provides services.

There are two types of servers:

- Web Server
- Application Server

#### Web Server

- Web server contains only web or servlet container. It can be used for servlet, jsp, struts, jsf etc. It can't be used for EJB.
- It is a computer where the web content can be stored. In general web server can be used to host the web sites but there also used some other web servers also such as FTP, email, storage, gaming etc.
- Examples of Web Servers are: **Apache Tomcat** and **Resin**.

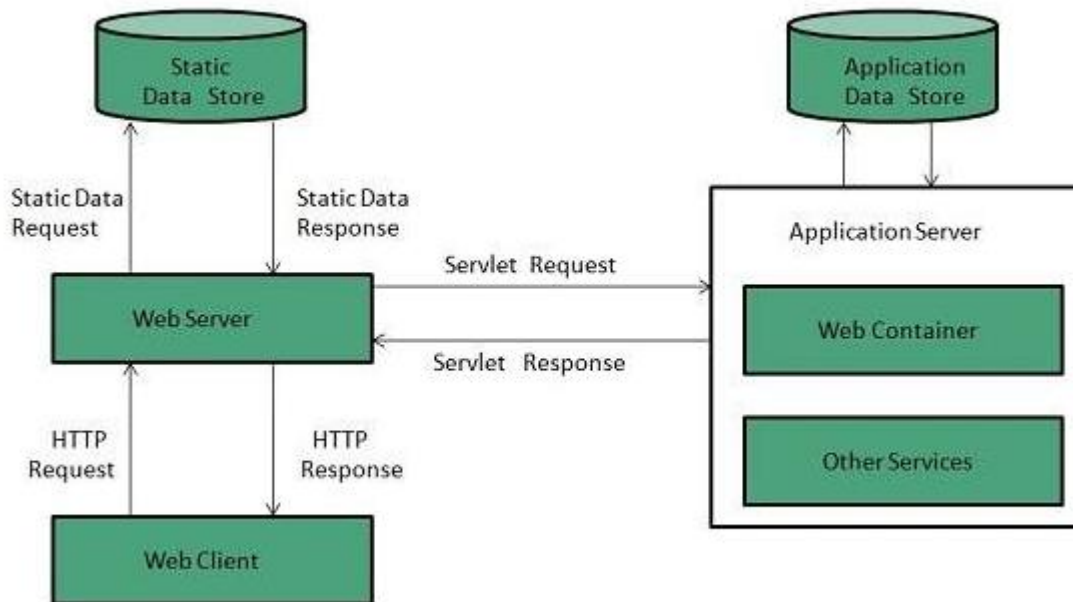


## Web Server Working

It can respond to the client request in either of the following two possible ways:

- Generating response by using the script and communicating with database.
- Sending file to the client associated with the requested URL.

The block diagram representation of Web Server is shown below:



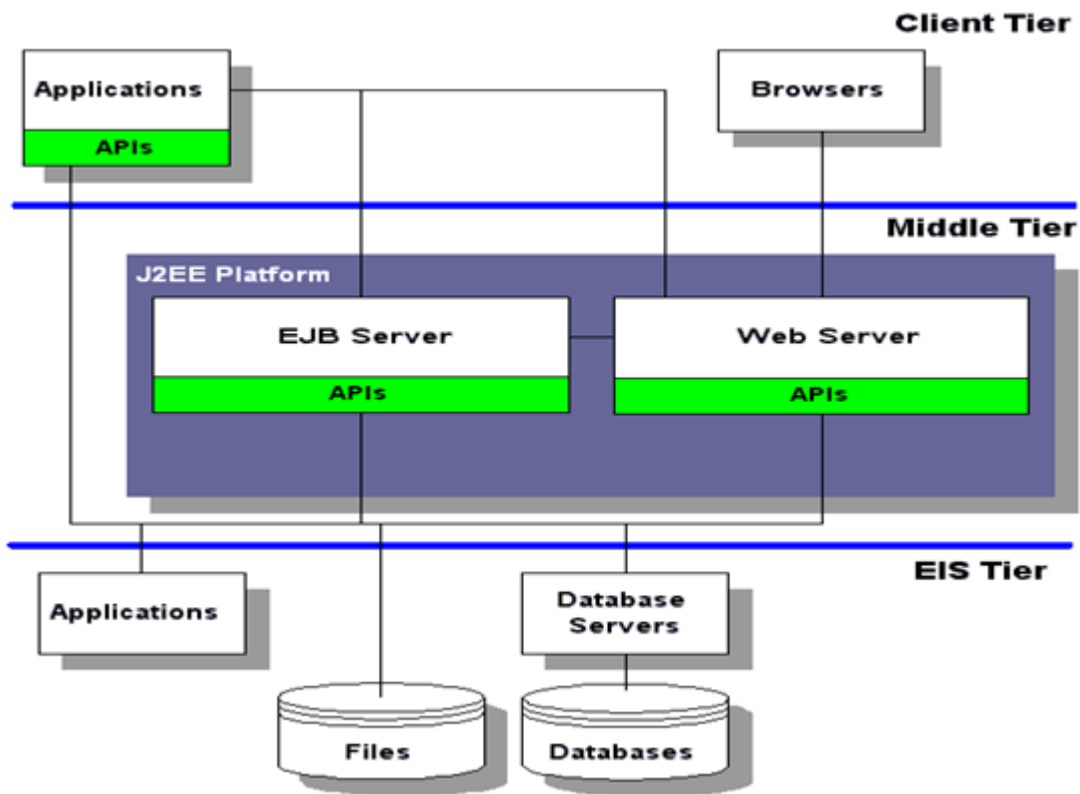
### Important points

- If the requested web page at the client side is not found, then web server will send the HTTP response: Error 404 Not found.
- When the web server searches for the requested page and it is found, it will send it to the client with an HTTP response.
- If the client requests some other resources, the web server will contact the application server, and data is stored for constructing the HTTP response.

### Application Server

- Application server contains Web and EJB containers.
- It can be used for servlet, jsp, struts, jsf, ejb etc. It is a component-based product that lies in the middle-tier of a server-centric architecture.
- It provides the middleware services for state maintenance and security, along with persistence and data access.
- It is a type of server designed to install, operate and host associated services and applications for the IT services, end users and organizations.

The block diagram representation of Application Server is shown below:



The Example of Application Servers are:

1. **JBoss:** Open-source server from JBoss community.
2. **Glassfish:** Provided by Sun Microsystem. Now acquired by Oracle.
3. **Weblogic:** Provided by Oracle. It more secured.
4. **Websphere:** Provided by IBM.

**Video Content / Details of website for further learning (if any):**

- <https://www.javatpoint.com/server-web-vs-application>
- <https://www.youtube.com/watch?v=BcmUOmv1N8>

**Important Books/Journals for further learning including the page nos.:**

- H.M. Deitel,P.J.Deitel,"Java How to Program" Prentice – Hall of India ,Newdelhi, Page No - 570

Course Faculty

Verified by HOD



# MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)

Rasipuram - 637 408, Namakkal Dist., Tamil Nadu



## LECTURE HANDOUTS

L - 23

MCA

I/II

Course Name with Code : 19CAB11 - INTERNET AND JAVA PROGRAMMING

Course Faculty : Mrs.G.Krishnaveni

Unit : PACKAGES

Date of Lecture: 20.03.2021

**Topic of Lecture:** Event Package

**Introduction :** ( Maximum 5 sentences)

- It is the root class from which all event state objects shall be derived. All Events are constructed with a reference to the object, the **source**, that is logically deemed to be the object upon which the Event in question initially occurred upon.
- This class is defined in java.util package.

**Prerequisite knowledge for Complete understanding and learning of Topic:**

- Packages and modules

**Detailed content of the Lecture:**

- Changing the state of an object is known as an event.
- For example, click on button, dragging mouse etc. The java.awt.event package provides many event classes and Listener interfaces for event handling.

Java Event classes and Listener interfaces

Event Classes	Listener Interfaces
ActionEvent	ActionListener
MouseEvent	MouseListener and MouseMotionListener
MouseWheelEvent	MouseWheelListener
KeyEvent	KeyListener
ItemEvent	ItemListener
TextEvent	TextListener

AdjustmentEvent	AdjustmentListener
WindowEvent	WindowListener
ComponentEvent	ComponentListener
ContainerEvent	ContainerListener
FocusEvent	FocusListener

### Registration Methods

For registering the component with the Listener, many classes provide the registration methods. For example:

- **Button**
  - public void addActionListener(ActionListener a){ }
- **MenuItem**
  - public void addActionListener(ActionListener a){ }
- **TextField**
  - public void addActionListener(ActionListener a){ }
  - public void addTextListener(TextListener a){ }
- **TextArea**
  - public void addTextListener(TextListener a){ }
- **Checkbox**
  - public void addItemListener(ItemListener a){ }
- **Choice**
  - public void addItemListener(ItemListener a){ }
- **List**
  - public void addActionListener(ActionListener a){ }
  - public void addItemListener(ItemListener a){ }

### Java event handling by implementing ActionListener

```
import java.awt.*;
import java.awt.event.*;
class AEvent extends Frame implements ActionListener{
    TextField tf;
    AEvent(){
```

```

//create components
tf=new TextField();
tf.setBounds(60,50,170,20);
Button b=new Button("click me");
b.setBounds(100,120,80,30);

//register listener
b.addActionListener(this);//passing current instance

//add components and set size, layout and visibility
add(b);add(tf);
setSize(300,300);
setLayout(null);
setVisible(true);
}
public void actionPerformed(ActionEvent e){
tf.setText("Welcome");
}
public static void main(String args[]){
new AEvent();
}
}

```

**public void setBounds(intxaxis, intyaxis, int width, int height);** have been used in the above example that sets the position of the component it may be button, textfield etc.



**Video Content / Details of website for further learning (if any):**

- <https://www.javatpoint.com/event-handling-in-java>
- <https://www.youtube.com/watch?v=CM7fnItrNr0>

**Important Books/Journals for further learning including the page nos.:**

- H.M. Deitel,P.J.Deitel,"Java How to Program" Prentice – Hall of India ,Newdelhi, Page No - 530

**Course Faculty**

**Verified by HOD**



# MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)

Rasipuram - 637 408, Namakkal Dist., Tamil Nadu



## LECTURE HANDOUTS

MCA

I/II

L - 24

Course Name with Code : 19CAB11 - INTERNET AND JAVA PROGRAMMING

Course Faculty : Mrs.G.Krishnaveni

Unit : PACKAGES

Date of Lecture: 22.03.2021

### Topic of Lecture: Event Model

#### Introduction : ( Maximum 5 sentences)

- The Delegation Event model is defined to handle events in GUI programming languages.
- The GUI stands for Graphical User Interface, where a user graphically/visually interacts with the system.

#### Prerequisite knowledge for Complete understanding and learning of Topic:

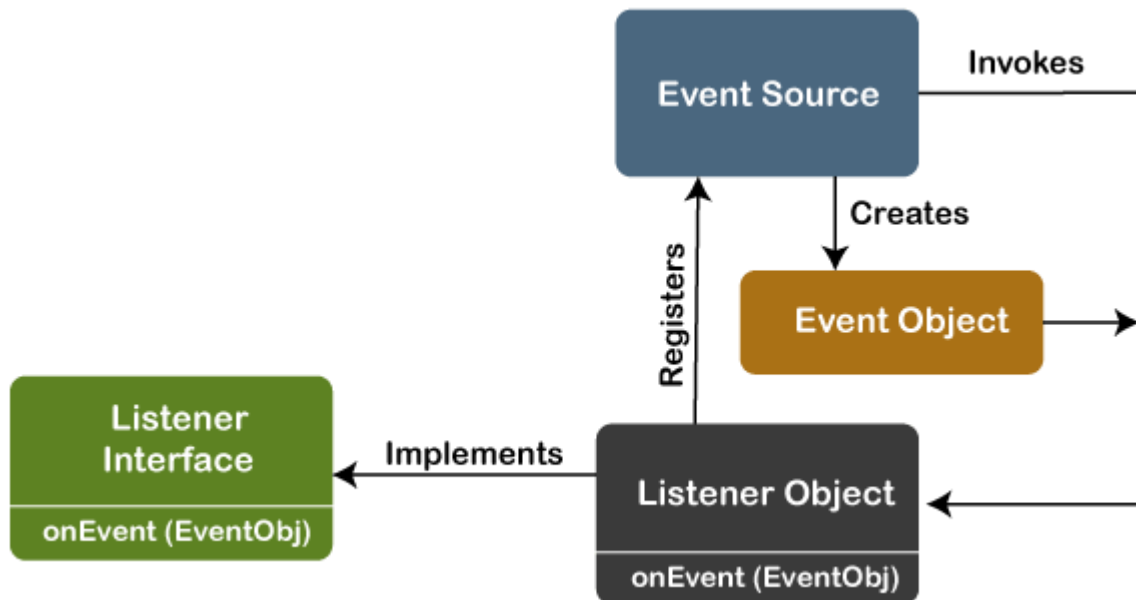
- Event handing/AWT

#### Detailed content of the Lecture:

- The GUI programming is inherently event-driven; whenever a user initiates an activity such as a mouse activity, clicks, scrolling, etc., each is known as an event that is mapped to a code to respond to functionality to the user. This is known as event handling.
- In this section, we will discuss event processing and how to implement the delegation event model in Java. We will also discuss the different components of an Event Model.

#### Event Processing in Java

- Java support event processing since Java 1.0. It provides support for AWT ( Abstract Window Toolkit), which is an API used to develop the Desktop application.
- In Java 1.0, the AWT was based on inheritance.
- To catch and process GUI events for a program, it should hold subclass GUI components and override action() or handleEvent() methods.
- The below image demonstrates the event processing



- But, the modern approach for event processing is based on the Delegation Model.
- It defines a standard and compatible mechanism to generate and process events.
- In this model, a source generates an event and forwards it to one or more listeners.
- The listener waits until it receives an event. Once it receives the event, it is processed by the listener and returns it.
- The UI elements are able to delegate the processing of an event to a separate function.
- The key advantage of the Delegation Event Model is that the application logic is completely separated from the interface logic.
- In this model, the listener must be connected with a source to receive the event notifications. Thus, the events will only be received by the listeners who wish to receive them. So, this approach is more convenient than the inheritance-based event model (in Java 1.0).
- In the older model, an event was propagated up the containment until a component was handled. This needed components to receive events that were not processed, and it took lots of time. The Delegation Event model overcame this issue.

Basically, an Event Model is based on the following three components:

- Events
- Events Sources
- Events Listeners

## Events

- The Events are the objects that define state change in a source.
- An event can be generated as a reaction of a user while interacting with GUI elements.
- Some of the event generation activities are moving the mouse pointer, clicking on a button, pressing the keyboard key, selecting an item from the list, and so on.
- We can also consider many other user operations as events.



- The Events may also occur that may be not related to user interaction, such as a timer expires, counter exceeded, system failures, or a task is completed, etc.
- We can define events for any of the applied actions.

### **Event Sources**

- A source is an object that causes and generates an event.
- It generates an event when the internal state of the object is changed.
- The sources are allowed to generate several different types of events.

### **Event Listeners**

- An event listener is an object that is invoked when an event triggers.
- The listeners require two things; first, it must be registered with a source; however, it can be registered with several resources to receive notification about the events.
- Second, it must implement the methods to receive and process the received notifications.

### **Types of Events**

The events are categories into the following two categories:

#### **The Foreground Events:**

- The foreground events are those events that require direct interaction of the user.
- These types of events are generated as a result of user interaction with the GUI component.
- For example, clicking on a button, mouse movement, pressing a keyboard key, selecting an option from the list, etc.

#### **The Background Events :**

- The Background events are those events that result from the interaction of the end-user.
- For example, an Operating system interrupts system failure (Hardware or Software).
- To handle these events, we need an event handling mechanism that provides control over the events and responses.

### **The Delegation Model**

- The Delegation Model is available in Java since Java 1.1.
- It provides a new delegation-based event model using AWT to resolve the event problems. It provides a convenient mechanism to support complex Java programs.

### **Design Goals**

The design goals of the event delegation model are as following:

- It is easy to learn and implement
- It supports a clean separation between application and GUI code.
- It provides robust event handling program code which is less error-prone (strong compile-

time checking)

- It is Flexible, can enable different types of application models for event flow and propagation.
- It enables run-time discovery of both the component-generated events as well as observable events.
- It provides support for the backward binary compatibility with the previous model.

**Video Content / Details of website for further learning (if any):**

- <https://www.javatpoint.com/delegation-event-model-in-java>
- <https://www.youtube.com/watch?v=hrh9-oIBr6A>

**Important Books/Journals for further learning including the page nos.:**

- H.M. Deitel,P.J.Deitel, ”Java How to Program” Prentice – Hall of India ,Newdelhi,  
Page No - 563

**Course Faculty**

**Verified by HOD**



# MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)

Rasipuram - 637 408, Namakkal Dist., Tamil Nadu



L - 25

## LECTURE HANDOUTS

MCA

I/II

Course Name with Code : 19CAB11 - INTERNET AND JAVA PROGRAMMING

Course Faculty : Mrs.G.Krishnaveni

Unit : PACKAGES

Date of Lecture: 23.03.2021

### Topic of Lecture: Garbage Collection

#### Introduction : ( Maximum 5 sentences)

- Garbage Collection is process of reclaiming the runtime unused memory automatically. In other words, it is a way to destroy the unused objects.

#### Prerequisite knowledge for Complete understanding and learning of Topic:

- Memory management/java

#### Detailed content of the Lecture:

- In java, garbage means unreferenced objects.
- Garbage Collection is process of reclaiming the runtime unused memory automatically. In other words, it is a way to destroy the unused objects.
- To do so, we were using free() function in C language and delete() in C++. But, in java it is performed automatically. So, java provides better memory management.

#### Advantage of Garbage Collection

- It makes java **memory efficient** because garbage collector removes the unreferenced objects from heap memory.
- It is **automatically done** by the garbage collector(a part of JVM) so we don't need to make extra efforts.

There are many ways:

- By nulling the reference
- By assigning a reference to another
- By anonymous object etc.

1) By nulling a reference:

```
Employee e=new Employee();  
e=null;
```

2) By assigning a reference to another:

```
Employee e1=new Employee();  
Employee e2=new Employee();  
e1=e2;//now the first object referred by e1 is available for garbage collection
```

3) By anonymous object:

```
new Employee();
```

### **finalize() method**

The finalize() method is invoked each time before the object is garbage collected. This method can be used to perform cleanup processing. This method is defined in Object class as:

```
protected void finalize(){ gc() method
```

The gc() method is used to invoke the garbage collector to perform cleanup processing. The gc() is found in System and Runtime classes.

```
public static void gc(){}
```

Simple Example of garbage collection in java

```
public class TestGarbage1 {  
    public void finalize(){System.out.println("object is garbage collected");}  
    public static void main(String args[]){  
        TestGarbage1 s1=new TestGarbage1();  
        TestGarbage1 s2=new TestGarbage1();  
        s1=null;  
        s2=null;  
        System.gc();  
    }  
}
```

**Video Content / Details of website for further learning (if any):**

- <https://www.javatpoint.com/Garbage-Collection>
- [https://www.youtube.com/watch?v=n\\_0gdyObLpc](https://www.youtube.com/watch?v=n_0gdyObLpc)

**Important Books/Journals for further learning including the page nos.:**

- H.M. Deitel,P.J.Deitel, "Java How to Program" Prentice – Hall of India ,Newdelhi, Page No - 1054

**Course Faculty**

**Verified by HOD**



# MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)  
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu



MCA

LECTURE HANDOUTS

I/II

L - 26

Course Name with Code : 19CAB11 - INTERNET AND JAVA PROGRAMMING

Course Faculty : Mrs.G.Krishnaveni

Unit : PACKAGES

Date of Lecture: 24.03.2021

## Topic of Lecture: Multi Threading

### Introduction : ( Maximum 5 sentences)

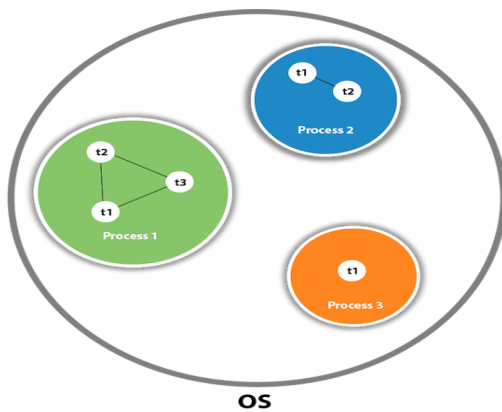
- A thread is a lightweight subprocess, the smallest unit of processing. It is a separate path of execution.
- Threads are independent. If there occurs exception in one thread, it doesn't affect other threads. It uses a shared memory area.

### Prerequisite knowledge for Complete understanding and learning of Topic:

- Memory management /processing

### Detailed content of the Lecture:

- A thread is a lightweight subprocess, the smallest unit of processing. It is a separate path of execution.
- Threads are independent. If there occurs exception in one thread, it doesn't affect other threads. It uses a shared memory area.



## Java Thread Methods

S.N.	Modifier and Type	Method	Description
1)	void	<code>start()</code>	It is used to start the execution of the thread.
2)	void	<code>run()</code>	It is used to do an action for a thread.
3)	static void	<code>sleep()</code>	It sleeps a thread for the specified amount of time.
4)	static Thread	<code>currentThread()</code>	It returns a reference to the currently executing object.
5)	void	<code>join()</code>	It waits for a thread to die.

### Life cycle of a Thread (Thread States)

In Java, a thread always exists in any one of the following states. These states are:

1. New
2. Active
3. Blocked / Waiting
4. Timed Waiting
5. Terminated

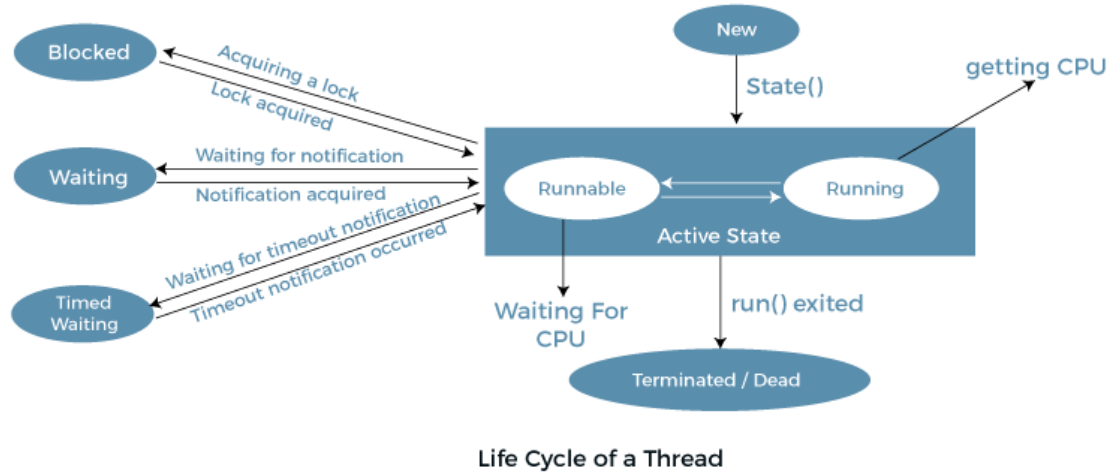
### Explanation of Different Thread States

**New:** Whenever a new thread is created, it is always in the new state. For a thread in the new state, the code has not been run yet and thus has not begun its execution.

**Active:** When a thread invokes the `start()` method, it moves from the new state to the active state. The active state contains two states within it: one is **runnable**, and the other is **running**.

- **Runnable:** A thread, that is ready to run is then moved to the runnable state. In the runnable state, the thread may be running or may be ready to run at any given instant of time. It is the duty of the thread scheduler to provide the thread time to run, i.e., moving the thread the running state.  
A program implementing multithreading acquires a fixed slice of time to each individual thread. Each and every thread runs for a short span of time and when that allocated time slice is over, the thread voluntarily gives up the CPU to the other thread, so that the other threads can also run for their slice of time. Whenever such a scenario occurs, all those threads that are willing to run, waiting for their turn to run, lie in the runnable state. In the runnable state, there is a queue where the threads lie.

- **Running:** When the thread gets the CPU, it moves from the runnable to the running state. Generally, the most common change in the state of a thread is from runnable to running and again back to runnable.



**Video Content / Details of website for further learning (if any):**

- <https://www.javatpoint.com/life-cycle-of-a-thread>

**Important Books/Journals for further learning including the page nos.:**

- H.M. Deitel,P.J.Deitel, ”Java How to Program” Prentice – Hall of India ,Newdelhi,
- Page No - 1055

**Course Faculty**

**Verified by HOD**



# MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)

Rasipuram - 637 408, Namakkal Dist., Tamil Nadu



## LECTURE HANDOUTS

MCA

I/II

L - 27

Course Name with Code : 19CAB11 - INTERNET AND JAVA PROGRAMMING

Course Faculty : Mrs.G.Krishnaveni

Unit : PACKAGES

Date of Lecture: 25.03.2021

### Topic of Lecture: Multithreading

#### Introduction : ( Maximum 5 sentences)

- **Multithreading in Java** is a process of executing multiple threads simultaneously.
- A thread is a lightweight sub-process, the smallest unit of processing. Multiprocessing and multithreading, both are used to achieve multitasking.

#### Prerequisite knowledge for Complete understanding and learning of Topic:

- Threading concepts/memory management

#### Detailed content of the Lecture:

- A thread is a lightweight sub-process, the smallest unit of processing. Multiprocessing and multithreading, both are used to achieve multitasking.
- However, we use multithreading than multiprocessing because threads use a shared memory area. They don't allocate separate memory area so saves memory, and context-switching between the threads takes less time than process.

Java Multithreading is mostly used in games, animation, etc.

#### Advantages of Java Multithreading

- 1) It **doesn't block the user** because threads are independent and you can perform multiple operations at the same time.
- 2) You **can perform many operations together, so it saves time.**
- 3) Threads are **independent**, so it doesn't affect other threads if an exception occurs in a single thread.

#### Multitasking

Multitasking is a process of executing multiple tasks simultaneously. We use multitasking to utilize the



CPU. Multitasking can be achieved in two ways:

- Process-based Multitasking (Multiprocessing)
- Thread-based Multitasking (Multithreading)

### 1) Process-based Multitasking (Multiprocessing)

- Each process has an address in memory. In other words, each process allocates a separate memory area.
- A process is heavyweight.
- Cost of communication between the process is high.
- Switching from one process to another requires some time for saving and loading [registers](#), memory maps, updating lists, etc.

### 2) Thread-based Multitasking (Multithreading)

- Threads share the same address space.
- A thread is lightweight.
- Cost of communication between the thread is low.

#### **EXAMPLES**

// Java code for thread creation by extending

// the Thread class

```
class MultithreadingDemo extends Thread {
    public void run()
    {
        try {
            // Displaying the thread that is running
            System.out.println(
                "Thread " + Thread.currentThread().getId()
                + " is running");
        }
        catch (Exception e) {
            // Throwing an exception
            System.out.println("Exception is caught");
        }
    }
}
```

// Main Class

```
public class Multithread {
    public static void main(String[] args)
    {
```

```
int n = 8; // Number of threads
for (inti = 0; i < n; i++) {
    MultithreadingDemo object
        = new MultithreadingDemo();
    object.start();
}
}
```

**Output**

Thread 15 is running  
Thread 14 is running  
Thread 16 is running  
Thread 12 is running  
Thread 11 is running  
Thread 13 is running  
Thread 18 is running  
Thread 17 is running

**Video Content / Details of website for further learning (if any):**

- <https://www.javatpoint.com/multithreading-in-java>
- [https://www.tutorialspoint.com/java/java\\_multithreading.htm](https://www.tutorialspoint.com/java/java_multithreading.htm)

**Important Books/Journals for further learning including the page nos.:**

- H.M. Deitel,P.J.Deitel,"Java How to Program" Prentice – Hall of India ,Newdelhi,
- Page No - 1057

**Course Faculty**

**Verified by HOD**



# MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)  
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu



## LECTURE HANDOUTS

L - 28

MCA

I/II

Course Name with Code : 19CAB11 & Internet And Java Programming

Course Faculty : Mrs.G.Krishnaveni

Unit : ADVANCED JAVA PROGRAMMING

Date of Lecture: 26.03.2021

### Topic of Lecture: Utility Packages

#### Introduction : ( Maximum 5 sentences)

- Package java.util contains the collections framework, legacy collection classes, event model, date and time facilities, internationalization, and miscellaneous utility classes (a string tokenizer, a random-number generator, and a bit array).
- Java package is used to categorize the classes and interfaces so that they can be easily maintained.

#### Prerequisite knowledge for Complete understanding and learning of Topic:

- Packages/AWT

#### Detailed content of the Lecture:

##### Java.util Package

- It contains the collections framework, legacy collection classes, event model, date and time facilities, internationalization, and miscellaneous utility classes (a string tokenizer, a random-number generator, and a bit array).

#### Following are the Important Classes in Java.util package :

- **AbstractCollection:** This class provides a skeletal implementation of the Collection interface, to minimize the effort required to implement this interface.
- **AbstractList:** This class provides a skeletal implementation of the List interface to minimize the effort required to implement this interface backed by a “random access” data store (such as an array).
- **AbstractMap<K,V>:** This class provides a skeletal implementation of the Map interface, to minimize the effort required to implement this interface.
- **AbstractMap.SimpleEntry<K,V>:** An Entry maintaining a key and a value.

- `AbstractMap.SimpleImmutableEntry<K,V>`: An Entry maintaining an immutable key and value.
- `AbstractQueue`: This class provides skeletal implementations of some Queue operations.
- `AbstractSequentialList`: This class provides a skeletal implementation of the List interface to minimize the effort required to implement this interface backed by a “sequential access” data store (such as a linked list).
- `AbstractSet`: This class provides a skeletal implementation of the Set interface to minimize the effort required to implement this interface.
- `ArrayDeque`: Resizable-array implementation of the Deque interface.
- [ArrayList](#): Resizable-array implementation of the List interface.
- [Arrays](#): This class contains various methods for manipulating arrays (such as sorting and searching).
- [BitSet](#): This class implements a vector of bits that grows as needed.
- `Calendar`: The Calendar class is an abstract class that provides methods for converting between a specific instant in time and a set of calendar fields such as YEAR, MONTH, DAY\_OF\_MONTH, HOUR, and so on, and for manipulating the calendar fields, such as getting the date of the next week.
- `Collections`: This class consists exclusively of static methods that operate on or return collections.
- `Currency`: Represents a currency.
- [Date](#): The class Date represents a specific instant in time, with millisecond precision.
- [Dictionary<K,V>](#): The Dictionary class is the abstract parent of any class, such as Hashtable, which maps keys to values.
- [EnumMap,V>](#): A specialized Map implementation for use with enum type keys.
- [EnumSet](#): A specialized Set implementation for use with enum types.
- `EventListenerProxy`: An abstract wrapper class for an EventListener class which associates a set of additional parameters with the listener.
- `EventObject`: The root class from which all event state objects shall be derived.
- `FormattableFlags`: FormattableFlags are passed to the `Formattable.formatTo()` method and modify the output format for Formattables.
- `Formatter`: An interpreter for printf-style format strings.
- `GregorianCalendar`: `GregorianCalendar` is a concrete subclass of `Calendar` and provides the standard calendar system used by most of the world.
- [HashMap<K,V>](#): Hash table based implementation of the Map interface.
- [HashSet](#): This class implements the Set interface, backed by a hash table (actually a `HashMap` instance).
- `Hashtable<K,V>`: This class implements a hash table, which maps keys to values.
- [IdentityHashMap<K,V>](#): This class implements the Map interface with a hash table, using reference-equality in place of object-equality when comparing keys (and values).
- [LinkedHashMap<K,V>](#): Hash table and linked list implementation of the Map interface, with predictable iteration order.
- [LinkedHashSet](#): Hash table and linked list implementation of the Set interface, with predictable iteration order.

- [LinkedList](#): Doubly-linked list implementation of the List and Deque interfaces.
- ListResourceBundle: ListResourceBundle is an abstract subclass of ResourceBundle that manages resources for a locale in a convenient and easy to use list.
- Locale – [Set 1](#), [Set 2](#): A Locale object represents a specific geographical, political, or cultural region.
- Locale.Builder: Builder is used to build instances of Locale from values configured by the setters.
- Objects: This class consists of static utility methods for operating on objects.
- Observable: This class represents an observable object, or “data” in the model-view paradigm.
- [PriorityQueue](#): An unbounded priority queue based on a priority heap.
- Properties: The Properties class represents a persistent set of properties.
- PropertyPermission: This class is for property permissions.
- PropertyResourceBundle: PropertyResourceBundle is a concrete subclass of ResourceBundle that manages resources for a locale using a set of static strings from a property file.
- [Random](#): An instance of this class is used to generate a stream of pseudorandom numbers.
- ResourceBundle: Resource bundles contain locale-specific objects.
- ResourceBundle.Control: ResourceBundle.Control defines a set of callback methods that are invoked by the ResourceBundle.getBundle factory methods during the bundle loading process.
- [Scanner](#): A simple text scanner which can parse primitive types and strings using regular expressions.
- ServiceLoader: A simple service-provider loading facility.
- SimpleTimeZone: SimpleTimeZone is a concrete subclass of TimeZone that represents a time zone for use with a Gregorian calendar.
- [Stack](#): The Stack class represents a last-in-first-out (LIFO) stack of objects.
- [StringTokenizer](#): The string tokenizer class allows an application to break a string into tokens.
- [Timer](#): A facility for threads to schedule tasks for future execution in a background thread.
- [TimerTask](#): A task that can be scheduled for one-time or repeated execution by a Timer.
- TimeZone: TimeZone represents a time zone offset, and also figures out daylight savings.
- TreeMap<K,V>: A Red-Black tree based NavigableMap implementation.
- [TreeSet](#): A NavigableSet implementation based on a TreeMap.
- WeakHashMap<K,V>: Hash table based implementation of the Map interface, with weak keys.

**Video Content / Details of website for further learning (if any):**

- [Java.util Package in Java - GeeksforGeeks](#)
- [Core Java Tutorial for Beginners #26 | Utility Classes in Java | java.util package - YouTube](#)

**Important Books/Journals for further learning including the page nos.:**

- H.M. Deitel,P.J.Deitel, ”Java How to Program” Prentice – Hall of India ,Newdelhi,
- Page No -1114

**Course Faculty**

**Verified by HOD**



# MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)  
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu



## LECTURE HANDOUTS

L - 29

MCA

I/II

Course Name with Code : 19CAB11 & Internet And Java Programming

Course Faculty : Mrs.G.Krishnaveni

Unit : ADVANCED JAVA PROGRAMMING

Date of Lecture: 27.03.2021

### Topic of Lecture: Utility Packages

#### Introduction : ( Maximum 5 sentences)

- Java.util package contains the collections framework, legacy collection classes, event model, date and time facilities, internationalization, and miscellaneous utility classes.
- This reference will take you through simple and practical methods available in java.util package.

#### Prerequisite knowledge for Complete understanding and learning of Topic:

- Packages/AWT

#### Detailed content of the Lecture:

Java Utility package is one of the most commonly used packages in the java program. The Utility Package of Java consist of the following components:

- **collections framework**
- **legacy collection classes**
- **event model**
- **date and time facilities**
- **internationalization**
- **miscellaneous utility classes** such as string tokenizer, random-number generator and bit array

Here are some of the description of the utility classes of this package:

- **Data Structure Classes**

Data Structure Classes are very useful classes for implementing standard computer science data

structures: including BitSet, Dictionary, Hashtable, Stack and Vector. The Enumeration interface of java.util package is used to count through a set of values.

## Date

The Date class is used to manipulate calendar dates in a system-independent fashion.

## StringTokenizer

This StringTokenizer class is used to convert a String of text into its tokens.

## Properties

The properties table contains key/value pairs where both the key and the value are Strings and the class is used by the System class to implement System properties.

## Observer and Observable

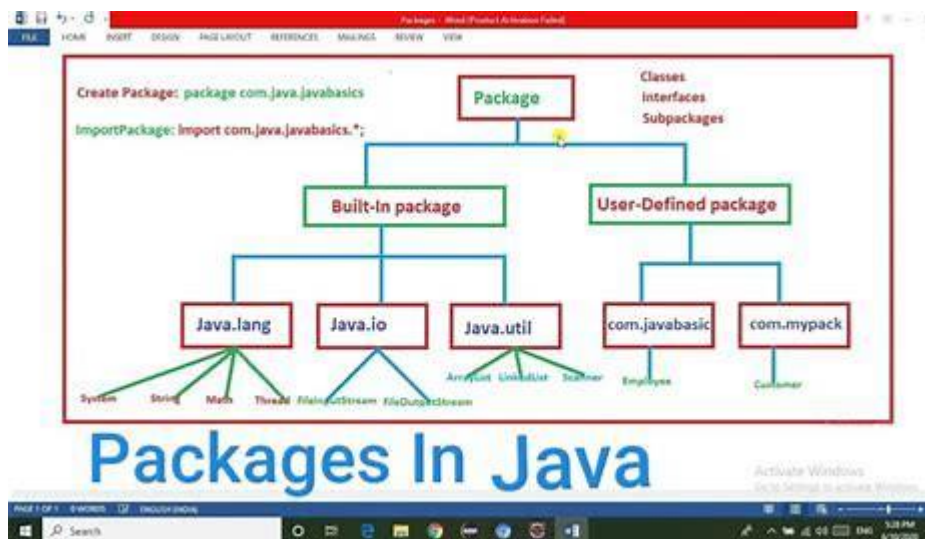
Classes that implement the Observer interface can "watch" Observable objects for state changes. When an Observable object changes it notifies all of its Observers of the change.

## Random-Number Generator

The Random Number Generator class is used to generate the random-numbers.

## Enumeration

The Enumeration interface defines a generic programming interface for iterating through a set of values.



## Video Content / Details of website for further learning (if any):

- [Java Util Package - Utility Package of Java \(roseindia.net\)](http://roseindia.net)
- [Java.util package tutorial \(tutorialspoint.com\)](http://tutorialspoint.com)
- [The java.util.concurrent Package \(Executors using Scala\) - YouTube](https://www.youtube.com/watch?v=...)

**Important Books/Journals for further learning including the page nos.:**

- H.M. Deitel,P.J.Deitel,"Java How to Program" Prentice – Hall of India ,Newdelhi,
- Page No -587

**Course Faculty**

**Verified by HOD**





# MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)  
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu



## LECTURE HANDOUTS

L - 30

MCA

I/II

Course Name with Code : 19CAB11 & Internet And Java Programming

Course Faculty : Mrs.G.Krishnaveni

Unit : ADVANCED JAVA PROGRAMMING

Date of Lecture: 29.03.2021

### Topic of Lecture: Input Output Packages

#### Introduction : ( Maximum 5 sentences)

- **Java I/O** (Input and Output) is used *to process the input and produce the output.*
- Java uses the concept of a stream to make I/O operation fast. The java.io package contains all the classes required for input and output operations.
- We can perform **file handling in Java** by Java I/O API.

#### Prerequisite knowledge for Complete understanding and learning of Topic:

- Packages/AWT

#### Detailed content of the Lecture:

##### Java I/O Tutorial

**Java I/O** (Input and Output) is used *to process the input and produce the output.*

Java uses the concept of a stream to make I/O operation fast. The java.io package contains all the classes required for input and output operations.

We can perform **file handling in Java** by Java I/O API.

##### Stream

A stream is a sequence of data. In Java, a stream is composed of bytes. It's called a stream because it is like a stream of water that continues to flow.

In Java, 3 streams are created for us automatically. All these streams are attached with the console.

1) **System.out**: standard output stream

2) **System.in**: standard input stream

3) **System.err**: standard error stream

Let's see the code to print **output and an error** message to the console.

1. `System.out.println("simple message");`
2. `System.err.println("error message");`

Let's see the code to get **input** from console.

1. `int i=System.in.read();//returns ASCII code of 1st character`
2. `System.out.println((char)i);//will print the character`

Do You Know?

- How to write a common data to multiple files using a single stream only?
- How can we access multiple files by a single stream?
- How can we improve the performance of Input and Output operation?
- How many ways can we read data from the keyboard?
- What does the console class?
- How to compress and uncompress the data of a file?

### *OutputStream vs InputStream*

The explanation of OutputStream and InputStream classes are given below:

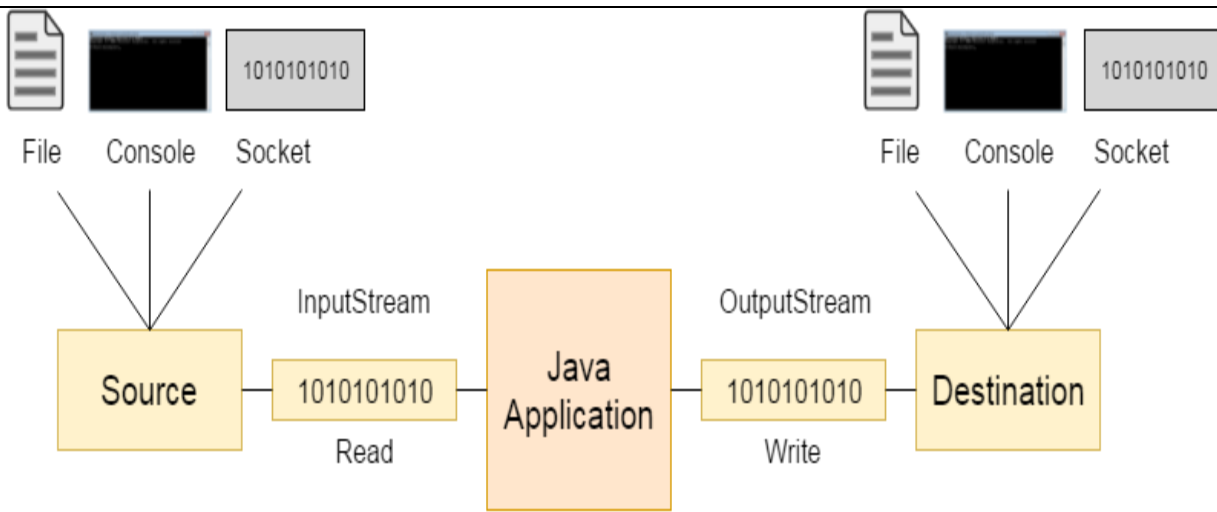
#### **OutputStream**

Java application uses an output stream to write data to a destination; it may be a file, an array, peripheral device or socket.

#### **InputStream**

Java application uses an input stream to read data from a source; it may be a file, an array, peripheral device or socket.

Let's understand the working of Java OutputStream and InputStream by the figure given below.



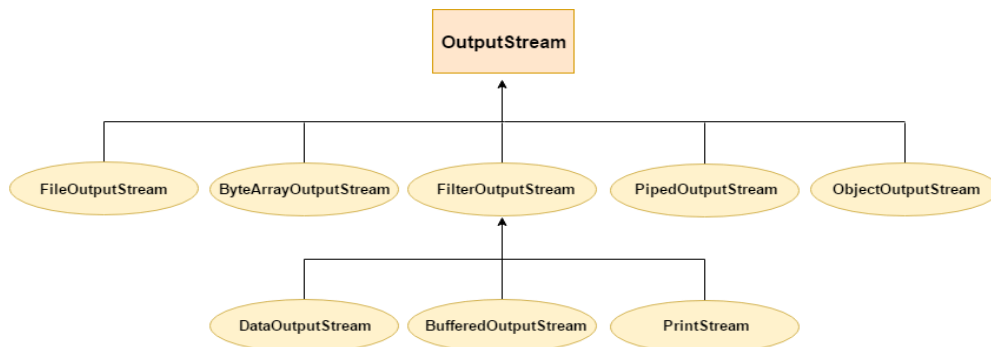
### *OutputStream class*

OutputStream class is an abstract class. It is the superclass of all classes representing an output stream of bytes. An output stream accepts output bytes and sends them to some sink.

### Useful methods of OutputStream

Method	Description
1) public void write(int) throws IOException	is used to write a byte to the current output stream.
2) public void write(byte[]) throws IOException	is used to write an array of byte to the current output stream.
3) public void flush() throws IOException	flushes the current output stream.
4) public void close() throws IOException	is used to close the current output stream.

### OutputStream Hierarchy



### *InputStream class*

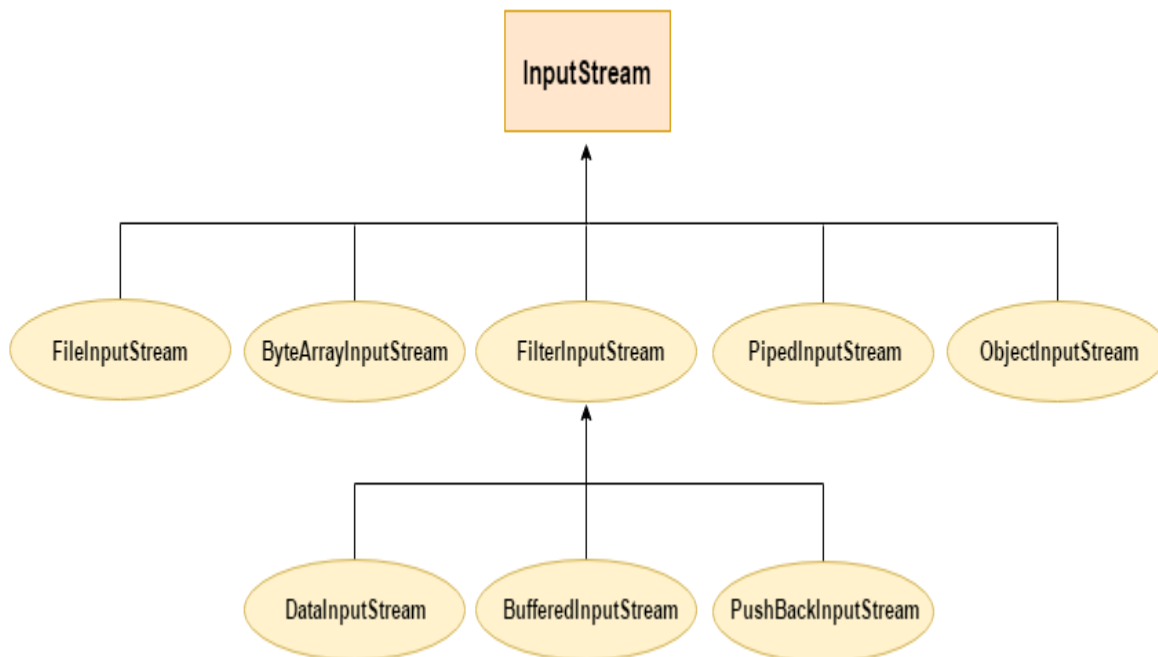
InputStream class is an abstract class. It is the superclass of all classes representing an input stream of

bytes.

### Useful methods of InputStream

Method	Description
1) public abstract int read()throws IOException	reads the next byte of data from the input stream. It returns -1 the end of the file.
2) public int available()throws IOException	returns an estimate of the number of bytes that can be read from the current input stream.
3) public void close()throws IOException	is used to close the current input stream.

### InputStream Hierarchy



### Video Content / Details of website for further learning (if any):

- <https://www.geeksforgeeks.org/java-io-packag/>

### Important Books/Journals for further learning including the page nos.:

- H.M. Deitel,P.J.Deitel,,"Java How to Program" Prentice – Hall of India ,Newdelhi,
- Page No -594

Course Faculty

Verified by HOD



## LECTURE HANDOUTS

L - 31

MCA

I/II

Course Name with Code : 19CAB11 & Internet And Java Programming

Course Faculty : Mrs.G.Krishnaveni

Unit : ADVANCED JAVA PROGRAMMING

Date of Lecture: 30.03.2021

### Topic of Lecture: Input Output Packages

#### Introduction : ( Maximum 5 sentences)


- The Java I/O package, a.k.a. java.io, provides a set of input streams and a set of output streams used to read and write data to files or other input and output sources. ...
- The pages of this lesson provide overviews of Java's I/O classes.

#### Prerequisite knowledge for Complete understanding and learning of Topic:

- Packages/AWT

#### Detailed content of the Lecture:

##### *The Java I/O Package*

- The Java I/O package, a.k.a. java.io, provides a set of input streams and a set of output streams used to read and write data to files or other input and output sources.
- There are three categories of classes in java.io: input streams, output streams and everything else.
- The pages of this lesson provide overviews of Java's I/O classes. They give information about what each class does and how you can use them. These pages do not provide any practical examples or details of each class. For more practical information regarding reading and writing data using these classes, see [Input and Output Streams](#) .

##### Input Streams

- Input streams read data from an input source. An input source can be a file, a string, or memory--anything that can contain data.
- All input streams inherit from InputStream--an abstract class that defines the programming interface for all input streams.
- The InputStream class defines a programming interface for reading bytes or arrays of bytes, marking locations in the stream, skipping bytes of input, finding out the number of bytes that are available for reading, and resetting the current position within the stream.
- An input stream is automatically opened when you create it. You can explicitly close a stream with the close() method, or let it be closed implicitly when the object is garbage collected.

## Output Streams

- Output streams write data to an output source. Similar to input sources, an output source can be anything that can contain data: a file, a string, or memory.
- The OutputStream class is a sibling to InputStream and is used to write data that can then be read by an input stream.
- The OutputStream class defines a programming interface for writing bytes or arrays of bytes to the stream and flushing the stream.
- Like an input stream, an output stream is automatically opened when you create it. You can explicitly close an output stream with the close() method, or let it be closed implicitly when the object is garbage collected.

### **How input is read from the Keyboard?**

- The “System.in” represents the keyboard. To read data from keyboard it should be connected to “InputStreamReader”. From the “InputStreamReader” it reads data from the keyboard and sends the data to the “BufferedReader”. From the “BufferedReader” it reads data from InputStreamReader and stores data in buffer. It has got methods so that data can be easily accessed.

### ***Reading Input from Console***

Input can be given either from file or keyword. In java, input can be read from console in 3 ways:

- BufferedReader
- StringTokenizer
- Scanner

### ***BufferedReader – Java class***

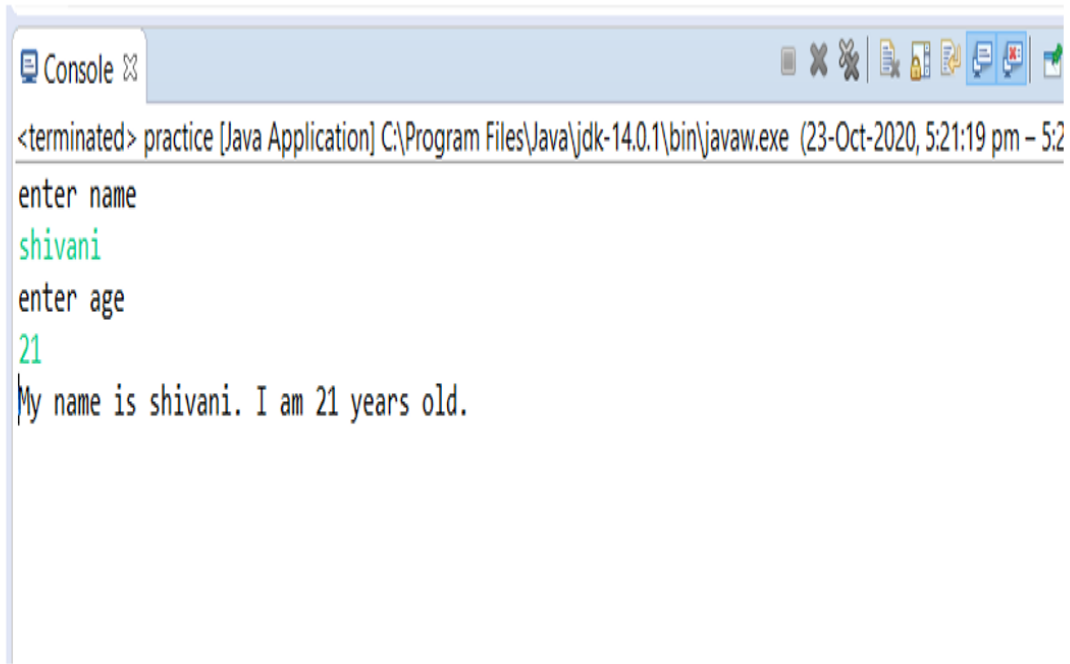
Here, we use the class “BufferedReader” and create the object “bufferedReader”. We also take integer value and fetch string from the user.

```
BufferedReader bufferedreader = new BufferedReader(new InputStreamReader(System.in));
int age = bufferedreader.read();
String name = bufferedreader.readLine();
```

From the eclipse window we also see an example of the same in the following code:

```
BufferedReader bufferedreader = new BufferedReader(
New InputStreamReader(System.in));
System.out.println(“enter name”);
String name = bufferedreader.readLine();
System.out.println(“enter age”);
int age = Integer.parseInt(bufferedReader.readLine());
int age1= bufferedreader.read();
System.out.println(“I am” + name + “ “+age+”years old”);
}
}
```

When we run the code in java ide, it will prompt the user to enter name and age as you can see below,



```
Console [X]
<terminated> practice [Java Application] C:\Program Files\Java\jdk-14.0.1\bin\javaw.exe (23-Oct-2020, 5:21:19 pm - 5:2
enter name
shivani
enter age
21
My name is shivani. I am 21 years old.
```

**Video Content / Details of website for further learning (if any):**

- [The Java I/O Package \(upv.es\)](http://upv.es)
- [Understanding Java Input and Output \(edureka.co\)](http://edureka.co)

**Important Books/Journals for further learning including the page nos.:**

- H.M. Deitel,P.J.Deitel,"Java How to Program" Prentice – Hall of India ,Newdelhi,
- Page No -597

**Course Faculty**

**Verified by HOD**



# MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)  
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu



## LECTURE HANDOUTS

L - 32

MCA

I/II

Course Name with Code : 19CAB11 & Internet And Java Programming

Course Faculty : Mrs.G.Krishnaveni

Unit : ADVANCED JAVA PROGRAMMING

Date of Lecture: 31.03.2021

**Topic of Lecture:** Inner Classes

**Introduction :** ( Maximum 5 sentences)

- In Java, it is also possible to nest classes (a class within a class).
- The purpose of nested classes is to group classes that belong together, which makes your code more readable and maintainable.

**Prerequisite knowledge for Complete understanding and learning of Topic:**

- Classes/Package/AWT

**Detailed content of the Lecture:**

In Java, inner class refers to the class that is declared inside class or interface which were mainly introduced, to sum up, same logically relatable classes as Java is purely object-oriented so bringing it closer to the real world. Now geeks you must be wondering why they were introduced?

**There are certain advantages associated with inner classes are as follows:**

- Making code clean and readable.
- Private methods of the outer class can be accessed, so bringing a new dimension and making it closer to the real world.
- Optimizing the code module.

*We do use them often as we go advance in java object-oriented programming where we want certain operations to be performed, granting access to limited classes and many more which will be clear as we do discuss and implement all types of inner classes in Java.*

### Types of Inner Classes

There are basically four types of inner classes in java.

1. Nested Inner Class
2. Method Local Inner Classes
3. Static Nested Classes
4. Anonymous Inner Classes

Let us discuss each of the above following types sequentially in-depth alongside a clean java program which is very crucial at every step as it becomes quite tricky as we adhere forwards.



### **Type 1: Nested Inner Class**

It can access any private instance variable of the outer class. Like any other instance variable, we can have access modifier private, protected, public, and default modifier. Like class, an interface can also be nested and can have access specifiers.

```
• // Java Program to Demonstrate Nested class
•
• // Class 1
• // Helper classes
• class Outer {
•
•     // Class 2
•     // Simple nested inner class
•     class Inner {
•
•         // show() method of inner class
•         public void show()
•         {
•
•             // Print statement
•             System.out.println("In a nested class method");
•         }
•     }
• }
•
• // Class 2
• // Main class
• class Main {
•
•     // Main driver method
•     public static void main(String[] args)
•     {
•
•         // Note how inner class object is created inside
•         // main()
•         Outer.Inner in = new Outer().new Inner();
•
•         // Calling show() method over above object created
•         in.show();
•     }
• }
```

**Output**

In a nested class method

**Video Content / Details of website for further learning (if any):**

- [Inner Class in Java - GeeksforGeeks](#)
- [Java Inner Class \(Nested Class\) \(w3schools.com\)](#)

**Important Books/Journals for further learning including the page nos.:**

- H.M. Deitel,P.J.Deitel,"Java How to Program" Prentice – Hall of India ,Newdelhi,
- Page No - 566

**Course Faculty**

**Verified by HOD**



# MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)

Rasipuram - 637 408, Namakkal Dist., Tamil Nadu



## LECTURE HANDOUTS

L - 33

MCA

I/II

**Course Name with Code : 19CAB11 & Internet And Java Programming**

**Course Faculty : Mrs.G.Krishnaveni**

**Unit : ADVANCED JAVA PROGRAMMING**

**Date of Lecture: 01.04.2021**

### Topic of Lecture: Java Database Connectivity

#### Introduction : ( Maximum 5 sentences)

- JDBC stands for Java Database Connectivity. JDBC is a Java API to connect and execute the query with the database.
- It is a part of JavaSE (Java Standard Edition). JDBC API uses JDBC drivers to connect with the database

#### Prerequisite knowledge for Complete understanding and learning of Topic:

- Class with database basic connection

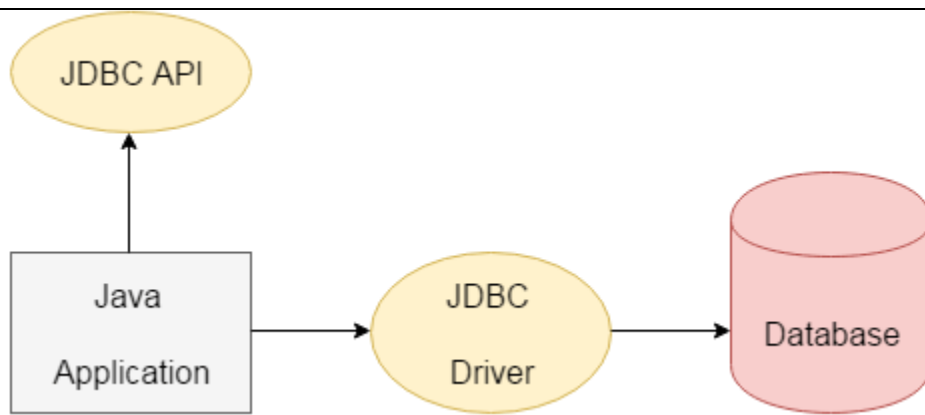
#### Detailed content of the Lecture:

JDBC stands for Java Database Connectivity. JDBC is a Java API to connect and execute the query with the database. It is a part of JavaSE (Java Standard Edition). JDBC API uses JDBC drivers to connect with the database. There are four types of JDBC drivers:

- JDBC-ODBC Bridge Driver,
- Native Driver,
- Network Protocol Driver, and
- Thin Driver

We have discussed the above four drivers in the next chapter.

- We can use JDBC API to access tabular data stored in any relational database.
- By the help of JDBC API, we can save, update, delete and fetch data from the database. It is like Open Database Connectivity (ODBC) provided by Microsoft.



The current version of JDBC is 4.3. It is the stable release since 21st September, 2017. It is based on the X/Open SQL Call Level Interface. The **java.sql** package contains classes and interfaces for JDBC API. A list of popular *interfaces* of JDBC API are given below:

- Driver interface
- Connection interface
- Statement interface
- PreparedStatement interface
- CallableStatement interface
- ResultSet interface
- ResultSetMetaData interface
- DatabaseMetaData interface
- RowSet interface

A list of popular *classes* of JDBC API are given below:

- DriverManager class
- Blob class
- Clob class
- Types class

### Why Should We Use JDBC

- Before JDBC, ODBC API was the database API to connect and execute the query with the database. But, ODBC API uses ODBC driver which is written in C language (i.e. platform dependent and unsecured). That is why Java has defined its own API (JDBC API) that uses JDBC drivers (written in Java language).

We can use JDBC API to handle database using Java program and can perform the following activities:

1. Connect to the database

2. Execute queries and update statements to the database
3. Retrieve the result received from the database.

### Do You Know

- How to connect Java application with Oracle and Mysql database using JDBC?
- What is the difference between Statement and PreparedStatement interface?
- How to print total numbers of tables and views of a database using JDBC?
- How to store and retrieve images from Oracle database using JDBC?
- How to store and retrieve files from Oracle database using JDBC?

### *What is API*

API (Application programming interface) is a document that contains a description of all the features of a product or software. It represents classes and interfaces that software programs can follow to communicate with each other. An API can be created for applications, libraries, operating systems, etc.

---

### *Topics in Java JDBC Tutorial*

#### 2) JDBC Drivers

In this JDBC tutorial, we will learn four types of JDBC drivers, their advantages and disadvantages.

---

#### 3) 5 Steps to connect to the Database

In this JDBC tutorial, we will see the five steps to connect to the database in Java using JDBC.

---

#### 4) Connectivity with Oracle using JDBC

In this JDBC tutorial, we will connect a simple Java program with the Oracle database.

---

#### 5) Connectivity with MySQL using JDBC

In this JDBC tutorial, we will connect a simple Java program with the MySQL database.

---

#### 6) Connectivity with Access without DSN

Let's connect java application with access database with and without DSN.

---

#### 7) DriverManager class

In this JDBC tutorial, we will learn what does the DriverManager class and what are its methods.

---

## 8) Connection interface

In this JDBC tutorial, we will learn what is Connection interface and what are its methods.

---

## 9) Statement interface

In this JDBC tutorial, we will learn what is Statement interface and what are its methods.

---

## 10) ResultSet interface

In this JDBC tutorial, we will learn what is ResultSet interface and what are its methods. Moreover, we will learn how we can make the ResultSet scrollable.

---

## 11) PreparedStatement Interface

In this JDBC tutorial, we will learn what is benefit of PreparedStatement over Statement interface. We will see examples to insert, update or delete records using the PreparedStatement interface.

---

## 12) ResultSetMetaData interface

In this JDBC tutorial, we will learn how we can get the metadata of a table.

---

## 13) DatabaseMetaData interface

In this JDBC tutorial, we will learn how we can get the metadata of a database.

---

## 14) Storing image in Oracle

Let's learn how to store image in the Oracle database using JDBC.

---

## 15) Retrieving image from Oracle

Let's see the simple example to retrieve image from the Oracle database using JDBC.

---

## 16) Storing file in Oracle

Let's see the simple example to store file in the Oracle database using JDBC.

---

## 17) Retrieving file from Oracle

Let's see the simple example to retrieve file from the Oracle database using JDBC.

---

### 18) CallableStatement

Let's see the code to call stored procedures and functions using CallableStatement.

---

### 19) Transaction Management using JDBC

Let's see the simple example to use transaction management using JDBC.

---

### 20) Batch Statement using JDBC

Let's see the code to execute batch of queries.

---

### 21) JDBC RowSet

Let's see the working of new JDBC RowSet interface.

---

#### **Video Content / Details of website for further learning (if any):**

- [JDBC Tutorial | What is Java Database Connectivity\(JDBC\) - javatpoint](#)
- <https://www.youtube.com/watch?v=5vzCjvUwMXg>

---

#### **Important Books/Journals for further learning including the page nos.:**

- H.M. Deitel,P.J.Deitel,"Java How to Program" Prentice – Hall of India ,Newdelhi, Page No - 792

**Course Faculty**

**Verified by HOD**



# MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)  
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu



## LECTURE HANDOUTS

L - 34

MCA

I/II

Course Name with Code : 19CAB11 & Internet And Java Programming

Course Faculty : Mrs.G.Krishnaveni

Unit : ADVANCED JAVA PROGRAMMING

Date of Lecture: 03.04.2021

**Topic of Lecture:** Servlets

**Introduction :** ( Maximum 5 sentences)

- **Servlet** technology is used to create a web application (resides at server side and generates a dynamic web page).
- **Servlet** technology is robust and scalable because of java language.
- Before Servlet, CGI (Common Gateway Interface) scripting language was common as a server-side programming language. However, there were many disadvantages to this technology.

**Prerequisite knowledge for Complete understanding and learning of Topic:**

- Classes and API / database

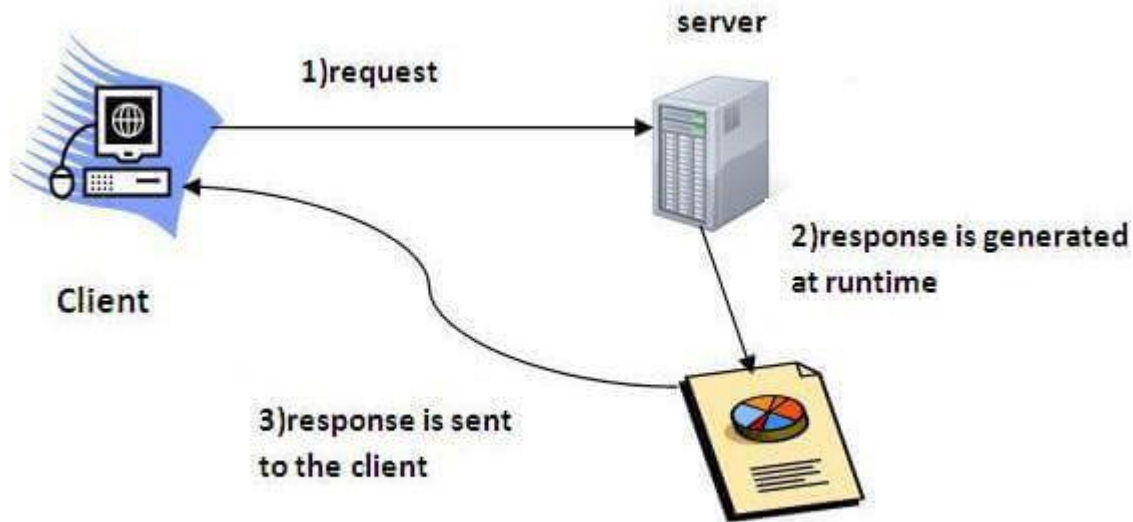
**Detailed content of the Lecture:**

Servlet can be described in many ways, depending on the context.

Keep Watching

- Servlet is a technology which is used to create a web application.
- Servlet is an API that provides many interfaces and classes including documentation.
- Servlet is an interface that must be implemented for creating any Servlet.
- Servlet is a class that extends the capabilities of the servers and responds to the incoming requests. It can respond to any requests.
- Servlet is a web component that is deployed on the server to create a dynamic web page.





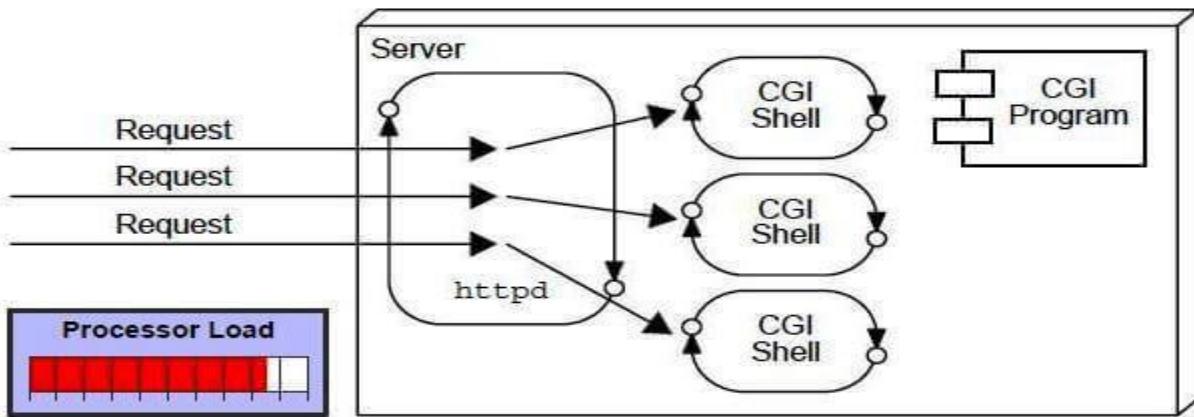
Do You Know?

- What is the web application and what is the difference between Get and Post request?
- What information is received by the web server if we request for a Servlet?
- How to run servlet in Eclipse, MyEclipse and Netbeans IDE?
- What is the difference between ServletConfig and ServletContext interface?
- How many ways can we maintain the state of a user? Which approach is mostly used in web development?
- How to count the total number of visitors and whole response time for a request using Filter?
- How to run servlet with annotation?
- How to create registration form using Servlet and Oracle database?
- How can we upload and download the file from the server?

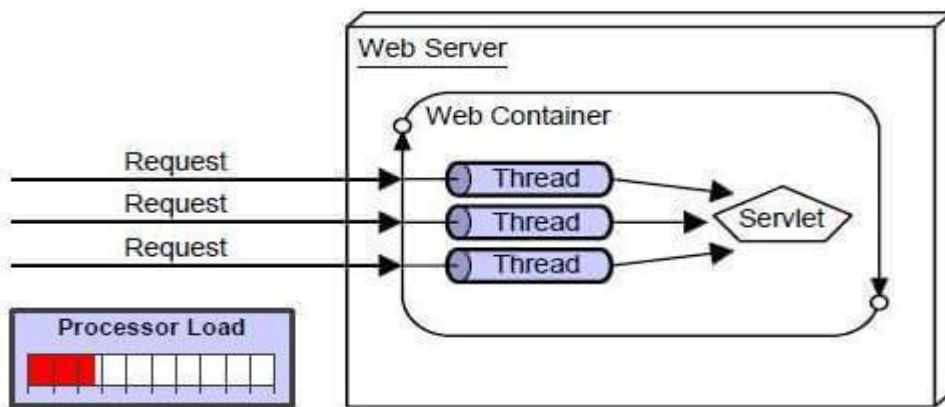
What is a web application?

- A web application is an application accessible from the web. A web application is composed of web components like Servlet, JSP, Filter, etc. and other elements such as HTML, CSS, and JavaScript. The web components typically execute in Web Server and respond to the HTTP request.
- CGI (Common Gateway Interface)

CGI technology enables the web server to call an external program and pass HTTP request information to the external program to process the request. For each request, it starts a new process.



### Advantages of Servlet



- There are many advantages of Servlet over CGI.
  - The web container creates threads for handling the multiple requests to the Servlet.
1. Threads have many benefits over the Processes such as they share a common memory area, lightweight, cost of communication between the threads are low. The advantages of Servlet are as follows:
    2. **Better performance:** because it creates a thread for each request, not process.
    3. **Portability:** because it uses Java language.
    4. **Robust:** JVM manages Servlets, so we don't need to worry about the memory leak, garbage collection, etc.
    5. **Secure:** because it uses java language.

### Servlets API's:

Servlets are build from two packages:

- javax.servlet(Basic)
- javax.servlet.http(Advance)

Various classes and interfaces present in these packages are:

Component	Type	Package
Servlet	Interface	javax.servlet.*
ServletRequest	Interface	javax.servlet.*
ServletResponse	Interface	javax.servlet.*
GenericServlet	Class	javax.servlet.*
HttpServlet	Class	javax.servlet.http.*
HttpServletRequest	Interface	javax.servlet.http.*
HttpServletResponse	Interface	javax.servlet.http.*
Filter	Interface	javax.servlet.*
ServletConfig	Interface	javax.servlet.*

**Video Content / Details of website for further learning (if any):**

- [Learn Servlet Tutorial - javatpoint](#)
- [Introduction to Java Servlets - GeeksforGeeks](#)

**Important Books/Journals for further learning including the page nos.:**

- H.M. Deitel,P.J.Deitel, ”Java How to Program” Prentice – Hall of India ,Newdelhi,  
Page No -854

**Course Faculty**

**Verified by HOD**



## LECTURE HANDOUTS

L - 35

MCA

I/II

Course Name with Code : 19CAB11 & Internet And Java Programming

Course Faculty : Mrs.G.Krishnaveni

Unit : ADVANCED JAVA PROGRAMMING

Date of Lecture: 05.04.2021

### Topic of Lecture: RMI

#### Introduction : ( Maximum 5 sentences)

- In computing, the **Java Remote Method Invocation** (Java RMI) is a Java API that performs remote method invocation
- the object-oriented equivalent of remote procedure calls (RPC), with support for direct transfer of serialized Java classes and distributed garbage-collection.

#### Prerequisite knowledge for Complete understanding and learning of Topic:

- classes / database RMI

#### Detailed content of the Lecture:

##### RMI (Remote Method Invocation)

1. [Remote Method Invocation \(RMI\)](#)
2. [Understanding stub and skeleton](#)
  1. [stub](#)
  2. [skeleton](#)
3. [Requirements for the distributed applications](#)
4. [Steps to write the RMI program](#)
5. [RMI Example](#)

- The **RMI** (Remote Method Invocation) is an API that provides a mechanism to create distributed application in java.
- The RMI allows an object to invoke methods on an object running in another JVM.
- The RMI provides remote communication between the applications using two objects *stub* and *skeleton*.
-

## Understanding stub and skeleton

RMI uses stub and skeleton object for communication with the remote object.

A **remote object** is an object whose method can be invoked from another JVM. Let's understand the stub and skeleton objects:

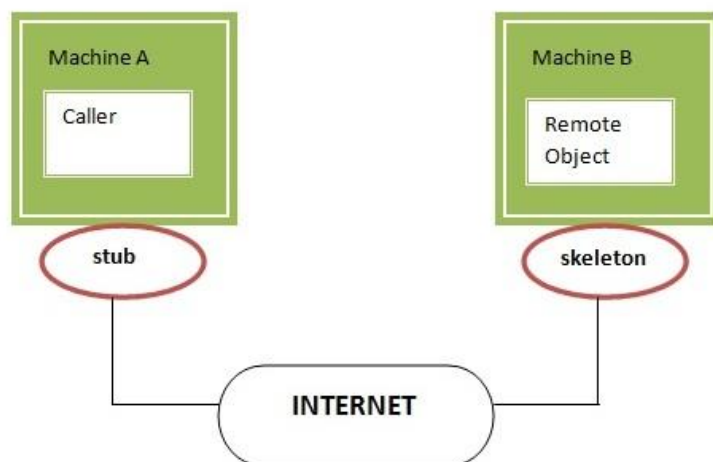
### stub

- The stub is an object, acts as a gateway for the client side. All the outgoing requests are routed through it. It resides at the client side and represents the remote object.
- When the caller invokes method on the stub object, it does the following tasks:
  1. It initiates a connection with remote Virtual Machine (JVM),
  2. It writes and transmits (marshals) the parameters to the remote Virtual Machine (JVM),
  3. It waits for the result
  4. It reads (unmarshals) the return value or exception, and
  5. It finally, returns the value to the caller.

### skeleton

- The skeleton is an object, acts as a gateway for the server side object.
- All the incoming requests are routed through it. When the skeleton receives the incoming request, it does the following tasks:
  1. It reads the parameter for the remote method
  2. It invokes the method on the actual remote object, and
  3. It writes and transmits (marshals) the result to the caller.

In the Java 2 SDK, an stub protocol was introduced that eliminates the need for



skeletons.

### *Understanding requirements for the distributed applications*

If any application performs these tasks, it can be distributed application.

1. The application need to locate the remote method
2. It need to provide the communication with the remote objects, and
3. The application need to load the class definitions for the objects.

The RMI application have all these features, so it is called the distributed application.

---

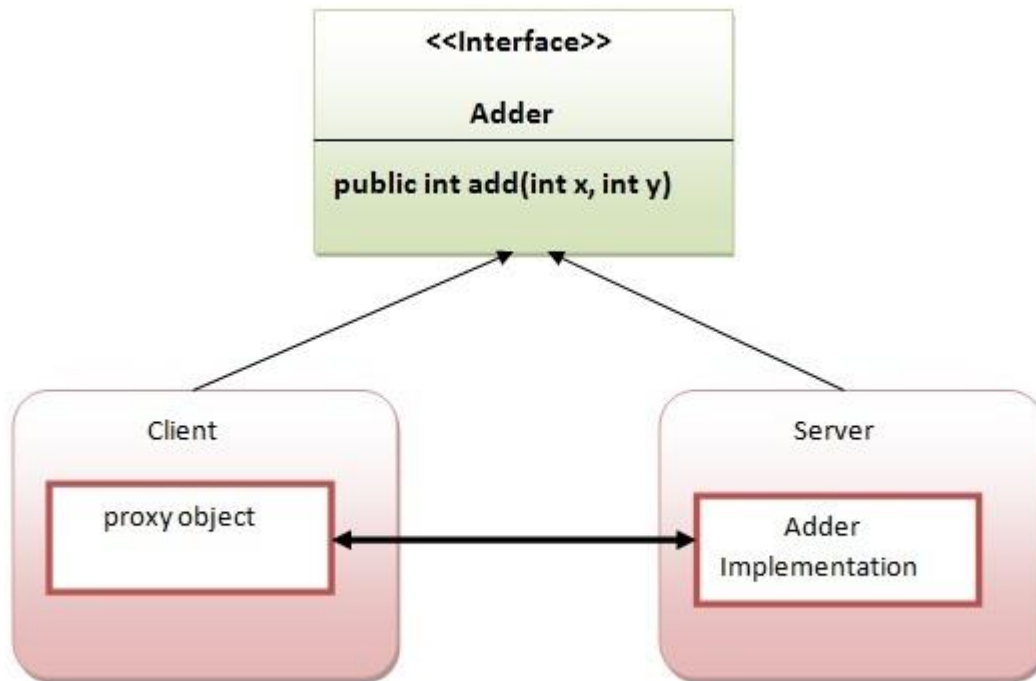
### *Java RMI Example*

The is given the 6 steps to write the RMI program.

1. Create the remote interface
  2. Provide the implementation of the remote interface
  3. Compile the implementation class and create the stub and skeleton objects using the rmic tool
  4. Start the registry service by rmiregistry tool
  5. Create and start the remote application
  6. Create and start the client application
- 

### *RMI Example*

In this example, we have followed all the 6 steps to create and run the rmi application. The client application need only two files, remote interface and client application. In the rmi application, both client and server interacts with the remote interface. The client application invokes methods on the proxy object, RMI sends the request to the remote JVM. The return value is sent back to the proxy object and then to the client application.



### 1) create the remote interface

For creating the remote interface, extend the Remote interface and declare the RemoteException with all the methods of the remote interface. Here, we are creating a remote interface that extends the Remote interface. There is only one method named add() and it declares RemoteException.

1. **import** java.rmi.\*;
2. **public interface** Adder **extends** Remote{
3. **public int** add(**int** x,**int** y)**throws** RemoteException;
4. }

### 2) Provide the implementation of the remote interface

Now provide the implementation of the remote interface. For providing the implementation of the Remote interface, we need to

- Either extend the UnicastRemoteObject class,
- or use the exportObject() method of the UnicastRemoteObject class

In case, you extend the UnicastRemoteObject class, you must define a constructor that declares RemoteException.

1. **import** java.rmi.\*;
2. **import** java.rmi.server.\*;
3. **public class** AdderRemote **extends** UnicastRemoteObject **implements** Adder{

4. AdderRemote()**throws** RemoteException{
5. **super**();
6. }
7. **public int** add(**int** x,**int** y){**return** x+y;}
8. }

### 3) create the stub and skeleton objects using the rmic tool.

Next step is to create stub and skeleton objects using the rmi compiler. The rmic tool invokes the RMI compiler and creates stub and skeleton objects.

1. rmic AdderRemote

### 4) Start the registry service by the rmiregistry tool

Now start the registry service by using the rmiregistry tool. If you don't specify the port number, it uses a default port number. In this example, we are using the port number 5000.

1. rmiregistry **5000**

### 5) Create and run the server application

Now rmi services need to be hosted in a server process. The Naming class provides methods to get and store the remote object. The Naming class provides 5 methods.

public static java.rmi.Remote lookup(java.lang.String) throws java.rmi.NotBoundException, java.net.MalformedURLException, java.rmi.RemoteException;	It returns the reference of the remote object.
public static void bind(java.lang.String, java.rmi.Remote) throws java.rmi.AlreadyBoundException, java.net.MalformedURLException, java.rmi.RemoteException;	It binds the remote object with the given name.
public static void unbind(java.lang.String) throws java.rmi.RemoteException, java.rmi.NotBoundException, java.net.MalformedURLException;	It destroys the remote object which is bound with the given name.
public static void rebind(java.lang.String, java.rmi.Remote) throws	It binds the remote object to the new name.



```
java.rmi.RemoteException,  
java.net.MalformedURLException;
```

```
public          static          java.lang.String[]  
list(java.lang.String)          throws  
java.rmi.RemoteException,  
java.net.MalformedURLException;
```

It returns an array of the names of the remote objects bound in the registry.

In this example, we are binding the remote object by the name sonoo.

1. **import** java.rmi.\*;
2. **import** java.rmi.registry.\*;
3. **public class** MyServer{
4. **public static void** main(String args[]){
5. **try**{
6. Adder stub=**new** AdderRemote();
7. Naming.rebind("rmi://localhost:5000/sonoo",stub);
8. }**catch**(Exception e){System.out.println(e);}
9. }
10. }

**Video Content / Details of website for further learning (if any):**

- [Remote Method Invocation \(RMI\) - javatpoint](#)
- <https://www.geeksforgeeks.org/remote-method-invocation-in-java/>

**Important Books/Journals for further learning including the page nos.:**

- Net reference

**Course Faculty**

**Verified by HOD**



# MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)

Rasipuram - 637 408, Namakkal Dist., Tamil Nadu



## LECTURE HANDOUTS

L - 36

MCA

I/II

Course Name with Code : 19CAB11 & Internet And Java Programming

Course Faculty : Mrs.G.Krishnaveni

Unit : ADVANCED JAVA PROGRAMMING

Date of Lecture: 06.04.2021

### Topic of Lecture: Swing Fundamentals

#### Introduction : ( Maximum 5 sentences)

- Swing in java is part of **Java foundation class** which is lightweight and platform independent.
- It is used for creating window based applications. It includes components like button, scroll bar, text field etc. Putting together all these components makes a graphical user interface.

#### Prerequisite knowledge for Complete understanding and learning of Topic:

- Classes / database swings

#### Detailed content of the Lecture:

- Java Swing tutorial is a part of Java Foundation Classes (JFC) that is *used to create window-based applications*.
- It is built on the top of AWT (Abstract Windowing Toolkit) API and entirely written in java.
- Unlike AWT, Java Swing provides platform-independent and lightweight components.

The javax.swing package provides classes for java swing API such as JButton, JTextField, JTextArea, JRadioButton, JCheckbox, JMenu, JColorChooser etc.

#### Difference between AWT and Swing

There are many differences between java awt and swing that are given below.

No.	Java AWT	Java Swing
1)	AWT components are <b>platform-dependent</b> .	Java swing components are <b>platform-independent</b> .

2)	AWT components are <b>heavyweight</b> .	Swing components are <b>lightweight</b> .
3)	AWT <b>doesn't support pluggable look and feel</b> .	Swing <b>supports pluggable look and feel</b> .
4)	AWT provides <b>less components</b> than Swing.	Swing provides <b>more powerful components</b> such as tables, lists, scrollpanes, colorchooser, tabbedPane etc.
5)	AWT <b>doesn't follows MVC</b> (Model View Controller) where model represents data, view represents presentation and controller acts as an interface between model and view.	Swing <b>follows MVC</b> .

### What is JFC

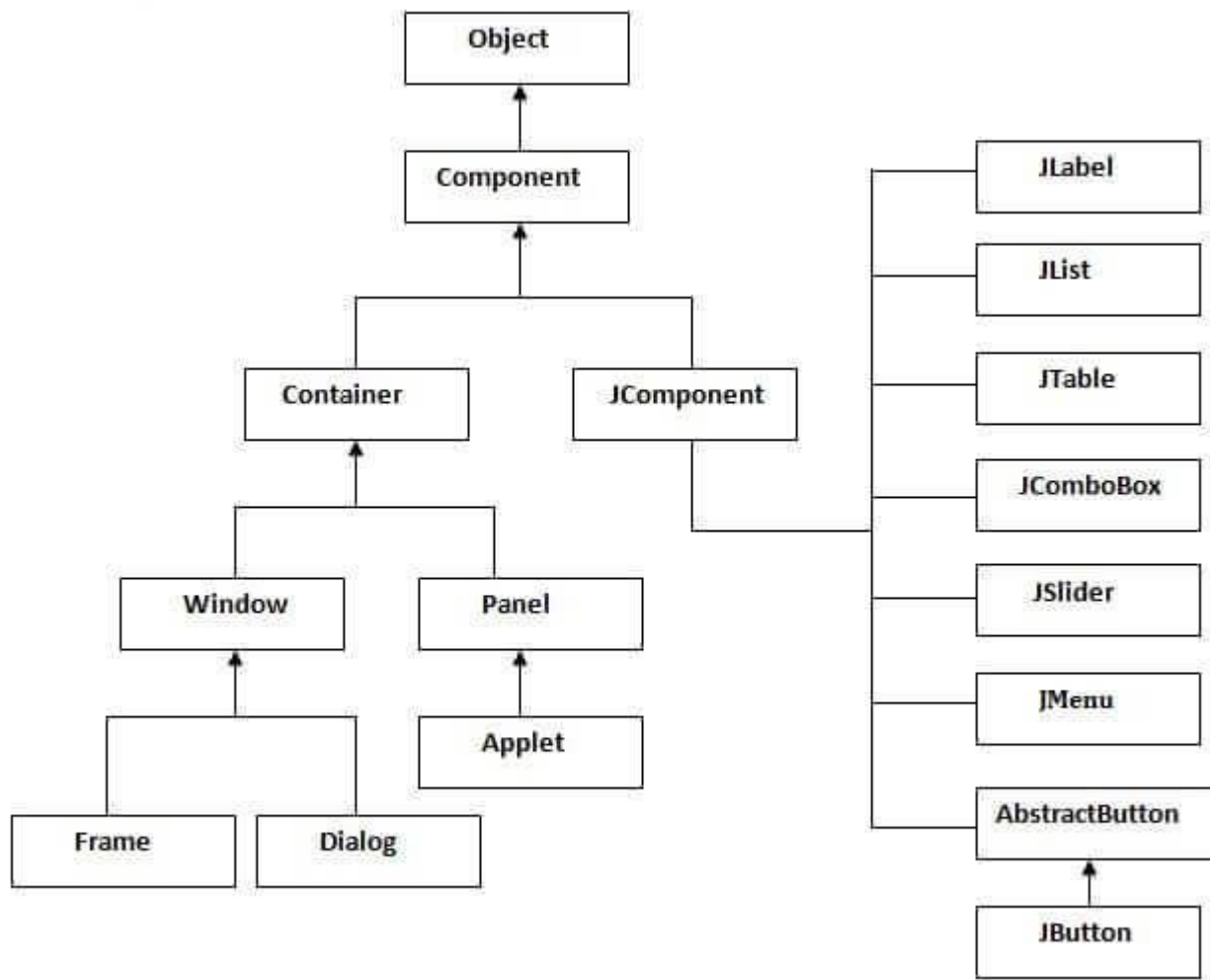
The Java Foundation Classes (JFC) are a set of GUI components which simplify the development of desktop applications.

### Do You Know

- How to create runnable jar file in java?
- How to display image on a button in swing?
- How to change the component color by choosing a color from ColorChooser ?
- How to display the digital watch in swing tutorial ?
- How to create a notepad in swing?
- How to create puzzle game and pic puzzle game in swing ?
- How to create tic tac toe game in swing ?

### Hierarchy of Java Swing classes

The hierarchy of java swing API is given below.



### Commonly used Methods of Component class

The methods of Component class are widely used in java swing that are given below.

Method	Description
public void add(Component c)	add a component on another component.
public void setSize(int width,int height)	sets size of the component.
public void setLayout(LayoutManager m)	sets the layout manager for the component.
public void setVisible(boolean b)	sets the visibility of the component. It is by default false.

### Java Swing Examples

There are two ways to create a frame:

- By creating the object of Frame class (association)
- By extending Frame class (inheritance)

We can write the code of swing inside the main(), constructor or any other method.

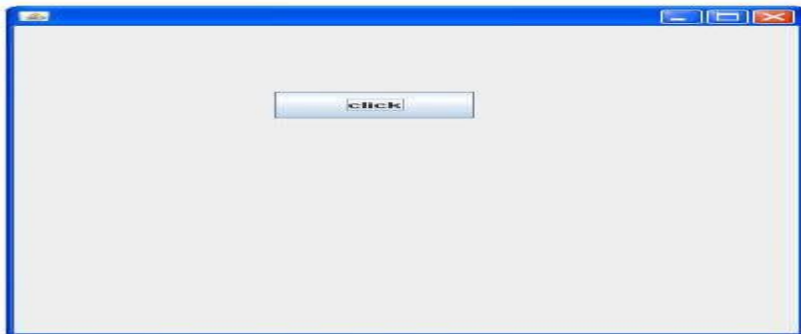
---

### Simple Java Swing Example

Let's see a simple swing example where we are creating one button and adding it on the JFrame object inside the main() method.

*File: FirstSwingExample.java*

1. **import** javax.swing.\*;
2. **public class** FirstSwingExample {
3. **public static void** main(String[] args) {
4. JFrame f=**new** JFrame();//creating instance of JFrame
- 5.
6. JButton b=**new** JButton("click");//creating instance of JButton
7. b.setBounds(130,100,100, 40);//x axis, y axis, width, height
- 8.
9. f.add(b);//adding button in JFrame
- 10.
11. f.setSize(400,500);//400 width and 500 height
12. f.setLayout(**null**);//using no layout managers
13. f.setVisible(**true**);//making the frame visible
14. }
15. }



---

### Example of Swing by Association inside constructor

We can also write all the codes of creating JFrame, JButton and method call inside the java constructor.

File: Simple.java

```
1. import javax.swing.*;
2. public class Simple {
3.     JFrame f;
4.     Simple(){
5.         f=new JFrame();//creating instance of JFrame
6.
7.         JButton b=new JButton("click");//creating instance of JButton
8.         b.setBounds(130,100,100, 40);
9.
10.        f.add(b);//adding button in JFrame
11.
12.        f.setSize(400,500);//400 width and 500 height
13.        f.setLayout(null);//using no layout managers
14.        f.setVisible(true);//making the frame visible
15.    }
16.
17.    public static void main(String[] args) {
18.        new Simple();
19.    }
20. }
```

**Video Content / Details of website for further learning (if any):**

- [Java Swing Tutorial - javatpoint](#)
- <https://www.youtube.com/watch?v=niJ1tANWwFk>

**Important Books/Journals for further learning including the page nos.:**

- **Net Reference**

**Course Faculty**

**Verified by HOD**



# MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)  
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu



## LECTURE HANDOUTS

L - 37

MCA

I/II

Course Name with Code : 19CAB11 & Internet And Java Programming

Course Faculty : Mrs.G.Krishnaveni

Unit : JAVA BEANS AND NETWORKING

Date of Lecture: 07.04.2021

### Topic of Lecture: Java Beans Application Builder Tools

#### Introduction : ( Maximum 5 sentences)

- The JavaBeans API makes it possible to write component software in the Java programming language.
- Components are self-contained, reusable software units that can be visually composed into composite components, applets, applications, and servlets using visual application builder tools.
- JavaBean components are known as *Beans*.

#### Prerequisite knowledge for Complete understanding and learning of Topic:

- Application Tools

#### Detailed content of the Lecture:

- Components expose their features (for example, public methods and events) to builder tools for visual manipulation.
- A Bean's features are exposed because feature names adhere to specific *design patterns*.
- A "JavaBeans-enabled" builder tool can then examine the Bean's patterns, discern its features, and expose those features for visual manipulation.
- A builder tool maintains Beans in a palette or toolbox.
- You can select a Bean from the toolbox, drop it into a form, modify its appearance and behavior, define its interaction with other Beans, and compose it and other Beans into an applet, application, or new Bean. All this can be done without writing a line of code.

#### The following list briefly describes key Bean concepts, and gives the chapter in the JavaBeans specification where you can read a complete description.

- Builder tools discover a Bean's features (that is, its properties, methods, and events) by a process known as *introspection*. Beans support introspection in two ways:
- By adhering to specific rules, known as *design patterns*, when naming Bean features. The [Introspector](#) class examines Beans for these design patterns to discover Bean features. The Introspector class relies on the [core reflection](#) API.

- The trail [The Reflection API](#) is an excellent place to learn about reflection.
  - By explicitly providing property, method, and event information with a related *Bean Information* class. A Bean information class implements the BeanInfo interface. A BeanInfo class explicitly lists those Bean features that are to be exposed to application builder tools.
- *Properties* are a Bean's appearance and behavior characteristics that can be changed at design time. Builder tools introspect on a Bean to discover its properties, and expose those properties for manipulation. Chapter 7 of the JavaBeans API Specification discusses properties.
- Beans expose properties so they can be *customized* at design time. Customization is supported in two ways: By using property editors, or by using more sophisticated Bean customizers. Chapter 9 of the JavaBeans API Specification discusses Bean customization.
- Beans use *events* to communicate with other Beans. A Bean that wants to receive events (a listener Bean) registers its interest with the Bean that fires the event (a source Bean). Builder tools can examine a Bean and determine which events that Bean can fire (send) and which it can handle (receive). Chapter 6 of the JavaBeans API Specification discusses events.
- *Persistence* enables Beans to save and restore their state. Once you've changed a Beans properties, you can save the state of the Bean and restore that Bean at a later time, property changes intact. JavaBeans uses Java Object Serialization to support persistence. Chapter 5 of the JavaBeans API Specification discusses persistence.
- A Bean's *methods* are no different than Java methods, and can be called from other Beans or a scripting environment. By default all public methods are exported.
- Although Beans are designed to be understood by builder tools, all key APIs, including support for events, properties, and persistence, have been designed to be easily read and understood by human programmers as well.
- JavaBeans is a software component architecture that extends the power of the Java language to enable well-formed objects to be manipulated visually in a builder tool such as Forte during design time.
- Such well-formed objects are referred to as *Java beans* or simply *beans*. The classes that define the beans, referred to as *JavaBeans components*, *bean components*, or simply *components*, must conform to the JavaBeans component model with the following:
  - A bean must be a public class.
  - A bean component must have a public default constructor (one that takes no arguments), although it can have other constructors, if needed. For example, a bean named MyBean either must have a constructor with the signature public MyBean(); or must have no constructor if its superclass has a default constructor.
  - A bean component must implement the Serializable interface to ensure a persistent state. JavaBeans can be used in a wide variety of tools, such as Lotus, Delphi, MS Visual Basic, and MS Word. Bean persistence may be required when JavaBeans are used in other tools.
  - Some tools are needed to save the beans and restore them later. Bean persistence ensures that the tools can reconstruct the properties and consistent behaviors of the bean to the state in



which it was saved.

- A bean component usually has properties with public accessor methods that enable them to be seen and updated visually by a builder tool. To enable the properties to be manipulated, the accessor methods must conform to the naming patterns or must be specified explicitly, using the BeanInfo interface. According to the accessor method-naming pattern, the method must be named get<PropertyName>() for getting the property value and set<PropertyName>() for setting the property value.
- A bean component may have events with public registration methods that enable it to add and remove listeners. If the bean plays a role as the source of events, it must provide registration methods.

**Video Content / Details of website for further learning (if any):**

- [JavaBeans Concepts \(iitk.ac.in\)](http://www.iitk.ac.in)
- [JavaBeans and Java Builder Tools | JavaBeans | InformIT](#)

**Important Books/Journals for further learning including the page nos.:**

- Net reference

**Course Faculty**

**Verified by HOD**



# MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)  
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu



## LECTURE HANDOUTS

L - 38

MCA

I/II

Course Name with Code : 19CAB11 & Internet And Java Programming

Course Faculty : Mrs.G.Krishnaveni

Unit : JAVA BEANS AND NETWORKING

Date of Lecture: 08.04.2021

### Topic of Lecture: Using the Bean Developer Kit

#### Introduction : ( Maximum 5 sentences)

- The Bean Developer Kit (BDK), available from the Java Soft site, is a simple example of a tool that enables you to create, configure, and connect a set of Beans.
- There is also a set of sample Beans with their source code. This section provides-step-by-step instructions for installing and using this tool.

#### Prerequisite knowledge for Complete understanding and learning of Topic:

- Classes & BDK

#### Detailed content of the Lecture:

##### The Bean Developer Kit (BDK)

- The Bean Developer Kit (BDK), available from the Java Soft site, is a simple example of a tool that enables you to create, configure, and connect a set of Beans. There is also a set of sample Beans with their source code. This section provides-step-by-step instructions for installing and using this tool.
- 'In this chapter, instructions are provided for a Windows 95/98/NT environment. The procedure» for a.UNIX platform are similar, some of the commands arc different.

##### Installing the BDK

- The JDK must be installed on your machine for the DDKto work. Confirm that the JOK tools are accessible from your environment.
- The'BDK can then be downloaded from the Java Soft site' (<http://java.sun.com>).Itis packaged as one file that is a self-extracting archive. Follow the instructions to install it on tour machine.

##### Starting the BDK

To start the DDK, follow these steps:

1. Change to the directory c:\bdk\beanbox.
2. Execute the batcl(·file called run.bat. This causes the BDK to display the three windows shown  
Toolbox lists all of the different Beans that have been included with the BDK. Bean Box provides an area to layout and connect the Beans selected from the Too Box. Properties .

## Using the BDK

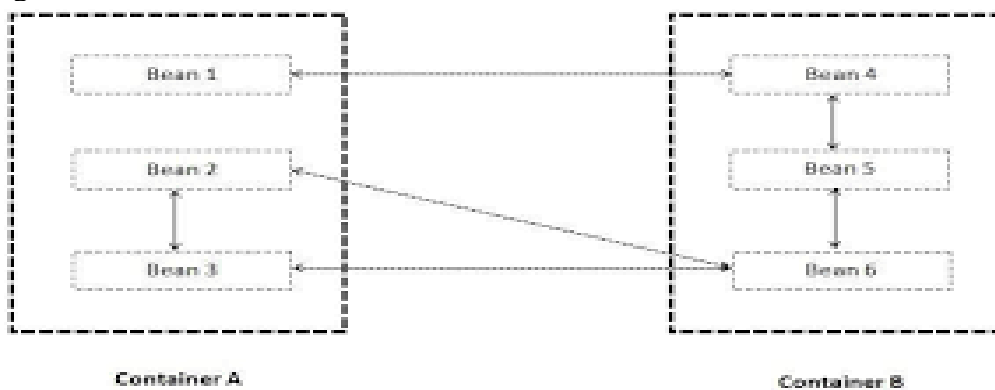
This section describes how to create an application by using some of the Beans provided with the BDK.

- First, the Molecule Bean displays at three-dimensional view of a molecule. It may be configured to present one of the following molecules: hydrochloric acid, benzene, counterintelligence, anticyclone, ethane, or water. This component also has methods that allow the molecule to be rotated in space along its X or Y axis.
- Second, the Our Bean provides a push-button functionality. We will have one.

## Create and Configure an Instance of the Molecule Bean

Follow these steps to create and configure an instance of the Molecule Bean.

1. Position the cursor on the Tool Box entry labeled Molecule and click the left mouse button. You should see the cursor change to a cross.
2. Move the cursor to the Bean Box display area and click the left mouse button in approximately the area where you wish the Bean to be displayed. You should see a rectangular region appear that contains a display of a molecule. This area is surrounded by a hatched border, indicating that it is currently selected.
3. You can reposition the Molecule Bean by positioning the cursor over one of the hatched borders and dragging the Bean.
4. You can change the molecule that is displayed by changing the selection in the Properties window. Notice that the Bean display changes immediately when you change the selected molecule.



A Sample Bean Interaction

## Video Content / Details of website for further learning (if any):

- [The Bean Developer Kit \(BDK\) Java Assignment Help, Online Java Project Help \(javahelponline.com\)](http://javahelponline.com)
- <https://slideplayer.com/slide/6584074/>

## Important Books/Journals for further learning including the page nos.:

- Net reference

Course Faculty

Verified by HOD



# MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)  
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu



## LECTURE HANDOUTS

L - 39

MCA

I/II

Course Name with Code : 19CAB11 & Internet And Java Programming

Course Faculty : Mrs.G.Krishnaveni

Unit : JAVA BEANS AND NETWORKING

Date of Lecture: 09.04.2021

### Topic of Lecture: Jar Files-Introspection- BDk

#### Introduction : ( Maximum 5 sentences)

- This is the process of **analyzing a Bean to determine its capabilities**. ... Without introspection, the Java Beans technology could not operate.
- There are two ways in which the developer of a Bean can indicate which of its properties, events, and methods should be exposed.

#### Prerequisite knowledge for Complete understanding and learning of Topic:

- Java beans and BDk

#### Detailed content of the Lecture:

##### Introspection

- At the core of Java Beans is *introspection*.
- This is the process of analyzing a Bean to determine its capabilities. This is an essential feature of the Java Beans API because it allows another application, such as a design tool, to obtain information about a component. Without introspection, the Java Beans technology could not operate.
- There are two ways in which the developer of a Bean can indicate which of its properties, events, and methods should be exposed.
- With the first method, simple naming conventions are used. These allow the introspection mechanisms to infer information about a Bean.
- In the second way, an additional class that extends the **BeanInfo** interface is provided that explicitly supplies this information. Both approaches are examined here.

##### Design Patterns for Properties

- A *property* is a subset of a Bean's state. The values assigned to the properties determine the

behavior and appearance of that component.

- A property is set through a *setter* method. A property is obtained by a *getter* method. There are two types of properties: simple and indexed.

### Simple Properties

- A simple property has a single value. It can be identified by the following design patterns, where **N** is the name of the property and **T** is its type:

```
public T getN( ) public void setN(T arg)
```

- A read/write property has both of these methods to access its values. A read-only property has only a get method. A write-only property has only a set method.
- Here are three read/write simple properties along with their getter and setter methods:

```
private double depth, height, width;
```

```
public double getDepth( ) { return depth;
```

```
}
```

```
public void setDepth(double d) { depth = d;
```

```
}
```

```
public double getHeight( ) { return height;
```

```
}
```

```
public void setHeight(double h) { height = h;
```

```
}
```

```
public double getWidth( ) { return width;
```

```
}
```

```
public void setWidth(double w) { width = w;
```

```
}
```

### Indexed Properties

- An indexed property consists of multiple values. It can be identified by the following design patterns, where **N** is the name of the property and **T** is its type:

```
public T getN(int index);
```

```
public void setN(int index, T value); public T[ ] getN( );
```

```
public void setN(T values[ ]);
```

Here is an indexed property called **data** along with its getter and setter methods:

```
private double data[ ];

public double getData(int index) { return data[index];

}

public void setData(int index, double value) { data[index] = value;

}

public double[ ] getData( ) { return data;

}

public void setData(double[ ] values) { data = new double[values.length];

System.arraycopy(values, 0, data, 0, values.length);

}
```

**Video Content / Details of website for further learning (if any):**

- [Introspection - Java Beans \(brainkart.com\)](http://brainkart.com)
- <https://www.youtube.com/watch?v=oUf8zKNIT0M>

**Important Books/Journals for further learning including the page nos.:**

- H.M. Deitel,P.J.Deitel,"Java How to Program" Prentice – Hall of India ,Newdelhi,
- Page No -1106

**Course Faculty**

**Verified by HOD**



# MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)  
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu



## LECTURE HANDOUTS

L - 40

MCA

I/II

Course Name with Code : 19CAB11 & Internet And Java Programming

Course Faculty : Mrs.G.Krishnaveni

Unit : JAVA BEANS AND NETWORKING

Date of Lecture: 10.04.2021

### Topic of Lecture: Using Bean Info Interface Persistence

#### Introduction : ( Maximum 5 sentences)

- A bean has the property of persistence when its properties, fields, and state information are saved to and retrieved from storage.
- Component models provide a mechanism for persistence that enables the state of components to be stored in a non-volatile place for later retrieval.

#### Prerequisite knowledge for Complete understanding and learning of Topic:

- Classes & java Bean

#### Detailed content of the Lecture:

##### Bean Persistence

- A bean has the property of persistence when its properties, fields, and state information are saved to and retrieved from storage.
- Component models provide a mechanism for persistence that enables the state of components to be stored in a non-volatile place for later retrieval.
- The mechanism that makes persistence possible is called *serialization*. Object serialization means converting an object into a data stream and writing it to storage.
- Any applet, application, or tool that uses that bean can then "reconstitute" it by deserialization. The object is then restored to its original state.

**For example**, a Java application can serialize a Frame window on a Microsoft Windows machine, the serialized file can be sent with e-mail to a Solaris machine, and then a Java application can restore the Frame window to the exact state which existed on the Microsoft Windows machine.

- Any applet, application, or tool that uses that bean can then "reconstitute" it by *deserialization*.
- All beans must persist. To persist, your beans must support serialization by implementing either the [java.io.Serializable](#) (in the API reference documentation) interface, or the [java.io.Externalizable](#) (in the API reference documentation) interface. These interfaces offer you the choices of automatic serialization and customized serialization. If any class in a class's

inheritance hierarchy implements Serializable or Externalizable, then that class is serializable.

### **Classes That Are Serializable**

- Any class is serializable as long as that class or a parent class implements the java.io.Serializable interface.
- Examples of serializable classes include Component, String, Date, Vector, and Hashtable.
- Thus, any subclass of the Component class, including Applet, can be serialized. Notable classes not supporting serialization include Image, Thread, Socket, and InputStream. Attempting to serialize objects of these types will result in a NotSerializableException.

The Java Object Serialization API automatically serializes most fields of a Serializable object to the storage stream. This includes primitive types, arrays, and strings. The API does not serialize or deserialize fields that are marked transient or static.

### **Controlling Serialization**

You can control the level of serialization that your beans undergo. Three ways to control serialization are:

- Automatic serialization, implemented by the Serializable interface. The Java serialization software serializes the entire object, except transient and static fields.
- Customized serialization. Selectively exclude fields you do not want serialized by marking with the transient (or static) modifier.
- Customized file format, implemented by the Externalizable interface and its two methods. Beans are written in a specific file format.

### **Default Serialization: The Serializable Interface**

- The Serializable interface provides automatic serialization by using the Java Object Serialization tools. Serializable declares no methods; it acts as a marker, telling the Object Serialization tools that your bean class is serializable.
- Marking your class Serializable means you are telling the Java Virtual Machine (JVM) that you have made sure your class will work with default serialization.
- Here are some important points about working with the Serializable interface:
  - Classes that implement Serializable must have an access to a *no-argument constructor* of supertype. This constructor will be called when an object is "reconstituted" from a .ser file.
  - You don't need to implement Serializable in your class if it is already implemented in a superclass.
  - All fields except static and transient fields are serialized. Use the transient modifier to specify fields you do not want serialized, and to specify classes that are not serializable.

### **Selective Serialization Using the transient Keyword**

- To exclude fields from serialization in a Serializable object mark the fields with the transient modifier.

Default serialization will not serialize transient and static fields.



## Selective Serialization: writeObject and readObject

- If your serializable class contains either of the following two methods (the signatures must be exact), then the default serialization will not take place.

```
private void writeObject(java.io.ObjectOutputStream out)
    throws IOException;
private void readObject(java.io.ObjectInputStream in)
    throws IOException, ClassNotFoundException;
```

- You can control how more complex objects are serialized, by writing your own implementations of the writeObject and readObject methods.
- Implement writeObject when you need to exercise greater control over what gets serialized when you need to serialize objects that default serialization cannot handle, or when you need to add data to the serialization stream that is not an object data member. Implement readObject to reconstruct the data stream you wrote with writeObject.

## The Externalizable Interface

- Use the Externalizable interface when you need complete control over your bean's serialization (for example, when writing and reading a specific file format). To use the Externalizable interface you need to implement two methods: readExternal and writeExternal. Classes that implement Externalizable must have a no-argument constructor.

```
public PropertyDescriptor[] getPropertyDescriptors() {
    try {
        PropertyDescriptor background =
            new PropertyDescriptor("background", beanClass);
        PropertyDescriptor foreground =
            new PropertyDescriptor("foreground", beanClass);
        PropertyDescriptor font =
            new PropertyDescriptor("font", beanClass);
        PropertyDescriptor label =
            new PropertyDescriptor("label", beanClass);

        background.setBound(true);
        foreground.setBound(true);
        font.setBound(true);
        label.setBound(true);

        PropertyDescriptor rv[] =
            {background, foreground, font, label};
        return rv;
    } catch (IntrospectionException e) {
        throw new Error(e.toString());
    }
}
```

**Video Content / Details of website for further learning (if any):**

- [Bean Persistence \(The Java™ Tutorials > JavaBeans\(TM\) > Advanced JavaBeans Topics\) \(oracle.com\)](#)
- [The BeanInfo Interface \(iitk.ac.in\)](#)

**Important Books/Journals for further learning including the page nos.:**

- Net reference

**Course Faculty**

**Verified by HOD**



# MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)

Rasipuram - 637 408, Namakkal Dist., Tamil Nadu



## LECTURE HANDOUTS

L - 41

MCA

I/II

**Course Name with Code : 19CAB11 & Internet And Java Programming**

**Course Faculty : Mrs.G.Krishnaveni**

**Unit : JAVA BEANS AND NETWORKING**

**Date of Lecture: 12.04.2021**

**Topic of Lecture:** Java Beans API

**Introduction : ( Maximum 5 sentences)**

- The JavaBeans API and its implementation are contained in the java.beans package. A few of the classes are used by beans while they run in an application.
- For example the event classes are used by beans that fire property and vetoable change events (see PropertyChangeEvent).

**Prerequisite knowledge for Complete understanding and learning of Topic:**

- Java beans & API

**Detailed content of the Lecture:**

- The JavaBeans API and its implementation are contained in the java.beans package. A few of the classes are used by beans while they run in an application.
- For example the event classes are used by beans that fire property and vetoable change events (see PropertyChangeEvent). However, most of the classes in this package are meant to be used by a bean editor that is, a development environment for customizing and putting together beans to create an application.
- In particular, these classes help the bean editor create a user interface that the user can use to customize the bean.
- For example, a bean may contain a property of a special type that a bean editor may not know how to handle. By using the PropertyEditor interface, a bean developer can provide an editor for this special type.
- The java.beans package provides support for long-term persistence - reading and writing a bean as a textual representation of its property values.
- The property values are treated as beans, and are recursively read or written to capture their publicly available state.
- This approach is suitable for long-term storage because it relies only on public API, rather than the likely-to-change private implementation.
- You read and write beans in XML format using the XMLDecoder and XMLEncoder classes, respectively. One notable feature of the persistence scheme is that reading in a bean requires no

special knowledge of the bean.

- To minimize the resources used by a bean, the classes used by bean editors are loaded only when the bean is being edited.
- They are not needed while the bean is running in an application and therefore not loaded. This information is kept in what's called a bean-info (see BeanInfo).
- These classes is not contained in core (real-time) library (rt.jar). You should use additional (design-time) library (dt.jar). This library contains bean-info classes for Swing.
- Package java.beans.beancontext provides classes and interfaces relating to bean context.
- A bean context is a container for beans and defines the execution environment for the beans it contains. There can be several beans in a single bean context, and a bean context can be nested within another bean context. This package also contains events and listener interface for beans being added and removed from a bean context.

### Future plans

**JSR 273: Design-Time API for JavaBeans** <<http://jcp.org/en/jsr/detail?id=273>>

This JSR extends the JavaBeans specification and APIs to improve design-time functionality for component authors to leverage within the visual design environments in IDEs.

**JSR 295: Beans Binding** <<http://jcp.org/en/jsr/detail?id=295>>

This JSR provides an API that allows two properties of two beans to stay in sync.

### Source Code & Building

The source code of the JavaBeans API and its implementation resides in the "src/share/classes" directory of the workspace. Java classes are located corresponding to the package hierarchy (eg java.beans.BeanInfo) are in "src/share/classes/java/beans/BeanInfo.java".

Non-public classes reside in:

- /src/share/classes/sun/beans

The makefile structure is as follows:

- make/java/beans - builds the JavaBeans API and its implementation in
- packages java.beans and java.beans.beancontext.
- make/sun/beans - builds default editors and bean-info classes.

```
package com.tutorialspoint;

public class StudentsBean implements java.io.Serializable {
    private String firstName = null;
    private String lastName = null;
    private int age = 0;

    public StudentsBean() {
    }
    public String getFirstName(){
        return firstName;
    }
    public String getLastName(){
        return lastName;
    }
}
```

```
}  
public int getAge(){  
    return age;  
}  
public void setFirstName(String firstName){  
    this.firstName = firstName;  
}  
public void setLastName(String lastName){  
    this.lastName = lastName;  
}  
public void setAge(Integer age){  
    this.age = age;  
}  
}
```

**Video Content / Details of website for further learning (if any):**

- [Beans \(java.net\)](http://java.net)
- [JSP - JavaBeans \(tutorialspoint.com\)](http://tutorialspoint.com)

**Important Books/Journals for further learning including the page nos.:**

- H.M. Deitel,P.J.Deitel,"Java How to Program" Prentice – Hall of India ,Newdelhi,
- Page No -243

**Course Faculty**

**Verified by HOD**



# MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)  
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu



## LECTURE HANDOUTS

L - 42

MCA

I/II

Course Name with Code : 19CAB11 & Internet And Java Programming

Course Faculty : Mrs.G.Krishnaveni

Unit : JAVA BEANS AND NETWORKING

Date of Lecture: 15.04.2021

### Topic of Lecture: Networking Basics Java and the Net

#### Introduction : ( Maximum 5 sentences)

- Java Networking is a concept of connecting two or more computing devices together so that we can share resources.
- Java socket programming provides facility to share data between different computing devices.

#### Prerequisite knowledge for Complete understanding and learning of Topic:

- Network basics

#### Detailed content of the Lecture:

##### Java Networking

Java Networking is a concept of connecting two or more computing devices together so that we can share resources.

Java socket programming provides facility to share data between different computing devices.

##### Advantage of Java Networking

1. Sharing resources
2. Centralize software management

#### Do You Know ?

- How to perform connection-oriented Socket Programming in networking ?
- How to display the data of any online web page ?
- How to get the IP address of any host name e.g. www.google.com ?
- How to perform connection-less socket programming in networking ?

The java.net package supports two protocols,

1. **TCP:** Transmission Control Protocol provides reliable communication between the sender and receiver. TCP is used along with the Internet Protocol referred as TCP/IP.
2. **UDP:** User Datagram Protocol provides a connection-less protocol service by allowing packet of data to be transferred along two or more nodes

### Java Networking Terminology

The widely used Java networking terminologies are given below:

1. IP Address
2. Protocol
3. Port Number
4. MAC Address
5. Connection-oriented and connection-less protocol
6. Socket

#### 1) IP Address

IP address is a unique number assigned to a node of a network e.g. 192.168.0.1 . It is composed of octets that range from 0 to 255.

It is a logical address that can be changed.

#### 2) Protocol

A protocol is a set of rules basically that is followed for communication. For example:

- TCP
- FTP
- Telnet
- SMTP
- POP etc.

#### 3) Port Number

The port number is used to uniquely identify different applications. It acts as a communication endpoint between applications.

The port number is associated with the IP address for communication between two applications.

#### 4) MAC Address

MAC (Media Access Control) address is a unique identifier of NIC (Network Interface Controller). A network node can have multiple NIC but each with unique MAC address.

For example, an ethernet card may have a **MAC** address of 00:0d:83::b1:c0:8e.

#### 5) Connection-oriented and connection-less protocol

In connection-oriented protocol, acknowledgement is sent by the receiver. So it is reliable but slow. The example of connection-oriented protocol is TCP.

But, in connection-less protocol, acknowledgement is not sent by the receiver. So it is not reliable but fast. The example of connection-less protocol is UDP.

#### 6) Socket

A socket is an endpoint between two way communications.

Visit next page for Java socket programming.

#### java.net package

The java.net package can be divided into two sections:

1. **A Low-Level API:** It deals with the abstractions of addresses i.e. networking identifiers, Sockets i.e. bidirectional data communication mechanism and Interfaces i.e. network interfaces.
2. **A High Level API:** It deals with the abstraction of URIs i.e. Universal Resource Identifier, URLs i.e. Universal Resource Locator, and Connections i.e. connections to the resource pointed by URLs.

The java.net package provides many classes to deal with networking applications in Java. A list of these classes is given below:

- Authenticator
- CacheRequest
- CacheResponse
- ContentHandler
- CookieHandler
- CookieManager
- DatagramPacket
- DatagramSocket
- DatagramSocketImpl



- InterfaceAddress
- JarURLConnection
- MulticastSocket
- InetSocketAddress
- InetAddress
- Inet4Address
- Inet6Address
- IDN
- HttpURLConnection
- HttpCookie
- NetPermission
- NetworkInterface
- PasswordAuthentication
- Proxy
- ProxySelector
- ResponseCache
- SecureCacheResponse
- ServerSocket
- Socket
- SocketAddress
- SocketImpl
- SocketPermission
- StandardSocketOptions
- URI
- URL
- URLClassLoader
- URLConnection
- URLEncoder
- URLEncoder
- URLStreamHandler

**List of interfaces available in java.net package:**

- ContentHandlerFactory
- CookiePolicy
- CookieStore
- DatagramSocketImplFactory
- FileNameMap
- SocketOption<T>
- SocketOptions
- SocketImplFactory
- URLStreamHandlerFactory
- ProtocolFamily

**Video Content / Details of website for further learning (if any):**

- [Java Networking - javatpoint](#)
- <https://www.youtube.com/watch?v=5Lm6nB3IsF8>

**Important Books/Journals for further learning including the page nos.:**

- H.M. Deitel,P.J.Deitel,"Java How to Program" Prentice – Hall of India ,Newdelhi,
- Page No -1107

**Course Faculty**

**Verified by HOD**



# MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)  
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu



## LECTURE HANDOUTS

L - 43

MCA

I/II

Course Name with Code : 19CAB11 & Internet And Java Programming

Course Faculty : Mrs.G.Krishnaveni

Unit : JAVA BEANS AND NETWORKING

Date of Lecture: 17.04.2021

### Topic of Lecture: Inet Addresses

#### Introduction : ( Maximum 5 sentences)

- The **java.net.InetAddress** class provides methods to get the IP address of any hostname.
- An IP address is represented by 32-bit or 128-bit unsigned number. InetAddress can handle both IPv4 and IPv6 addresses.

#### Prerequisite knowledge for Complete understanding and learning of Topic:

- Network Basics & java Inet

#### Detailed content of the Lecture:

**public class InetAddress** extends **Object** implements **Serializable**:

The **java.net.InetAddress** class provides methods to get the IP address of any hostname. An IP address is represented by 32-bit or 128-bit unsigned number. InetAddress can handle both IPv4 and IPv6 addresses.

There are 2 types of addresses :

1. **Unicast** — An identifier for a single interface.
2. **Multicast** — An identifier for a set of interfaces.

#### InetAddress – Factory Methods :

The **InetAddress** class is used to encapsulate both, the numerical IP address and the domain name for that address. The InetAddress class has no visible constructors. The InetAddress class has the inability to create objects directly, hence *factory methods* are used for the purpose. Factory Methods are static methods in a class that return an object of that class.

There are 5 factory methods available in InetAddress class –

Method	Description
static InetAddress getLocalHost() <i>throws UnknownHostException</i>	This method returns the instance of InetAddress containing the local hostname and address.
public static InetAddress getByName( String host ) <i>throws UnknownHostException</i>	This method returns the instance of InetAddress containing LocalHost IP and name.
static InetAddress[] getAllByName( String hostName ) <i>throws UnknownHostException</i>	This method returns the array of the instance of InetAddress class which contains IP addresses.
static InetAddress getByAddress( byte IPAddress[] ) <i>throws UnknownHostException</i>	This method returns an InetAddress object created from the raw IP address.
static InetAddress getByAddress( String hostName, byte IPAddress[] ) <i>throws UnknownHostException</i>	This method creates and returns an InetAddress based on the provided hostname and IP address.

Below is the Java implementation of InetAddress class to demonstrate the use of factory methods –

- Java

```

import java.io.*;
import java.net.*;
import java.util.*;

class GFG {
    public static void main(String[] args)
        throws UnknownHostException
    {
        // To get and print InetAddress of Local Host
        InetAddress address1 = InetAddress.getLocalHost();
        System.out.println("InetAddress of Local Host : "
            + address1);

        // To get and print InetAddress of Named Host
        InetAddress address2
            = InetAddress.getByName("45.22.30.39");
        System.out.println("InetAddress of Named Host : "
            + address2);

        // To get and print ALL InetAddresses of Named Host
        InetAddress address3[]

```

```

    = InetAddress.getAllByName("172.19.25.29");
for (int i = 0; i < address3.length; i++) {
    System.out.println(
        "ALL InetAddresses of Named Host : "
        + address3[i]);
    }

// To get and print InetAddresses of
// Host with specified IP Address
byte IPAddress[] = { 125, 0, 0, 1 };
InetAddress address4
    = InetAddress.getByAddress(IPAddress);
System.out.println(
    "InetAddresses of Host with specified IP Address : "
    + address4);

// To get and print InetAddresses of Host
// with specified IP Address and hostname
byte[] IPAddress2
    = { 105, 22, (byte)223, (byte)186 };
InetAddress address5 = InetAddress.getByAddress(
    "gfg.com", IPAddress2);
System.out.println(
    "InetAddresses of Host with specified IP Address and hostname : "
    + address5);
}
}

```

### Output

InetAddress of Local Host : localhost/127.0.0.1

InetAddress of Named Host : /45.22.30.39

ALL InetAddresses of Named Host : /172.19.25.29

InetAddresses of Host with specified IP Address : /125.0.0.1

InetAddresses of Host with specified IP Address and hostname : gfg.com/105.22.223.186

### InetAddress — Instance Methods :

InetAddress class has plenty of instance methods that can be called using the object. The instance methods are –

Method	Description
equals(Object obj)	This function compares this object against the specified object.

<code>getAddress()</code>	This method returns the raw IP address of this <code>InetAddress</code> object, in bytes.
<code>getCanonicalHostName()</code>	This method returns the fully qualified domain name for this IP address.
<code>getHostAddress()</code>	This method gets the IP address in string form.
<code>getHostName()</code>	This method returns the host name for this IP address.
<code>hashCode()</code>	This method gets a hashcode for this IP address.
<code>isAnyLocalAddress()</code>	This method utility routine to check if the <code>InetAddress</code> is an unpredictable address.
<code>isLinkLocalAddress()</code>	This method utility routine to check if the address is not linked to local unicast address.
<code>isLoopbackAddress()</code>	This method used to check if the <code>InetAddress</code> represents a loopback address.
<code>isMCGlobal()</code>	This method utility routine check if this address has a multicast address of global scope.
<code>isMCLinkLocal()</code>	This method utility routine check if this address has a multicast address of link-local scope.
<code>isMCNodeLocal()</code>	This method utility routine check if the multicast address has node scope.
<code>isMCOrgLocal()</code>	This method utility routine check if the multicast address has an organization scope.
<code>isMCSiteLocal()</code>	This method utility routine check if the multicast address has site scope.
<code>isMulticastAddress()</code>	This method checks whether the site has multiple servers.
<code>isReachable(int timeout)</code>	This method tests whether that address is

reachable.

isReachable(NetworkInterface netif, int  
ttl, int timeout)

This method tests whether that address is  
reachable.

isSiteLocalAddress()

This method utility routine check if the  
InetAddress is a site-local address.

toString()

This method converts and returns an IP address  
in string form.

Below is the Java implementation of InetAddress class to demonstrate the use of instance methods –

- Java

```
import java.io.*;
import java.net.*;
import java.util.*;

class GFG {
    public static void main(String[] args)
        throws UnknownHostException
    {

        InetAddress address1
            = InetAddress.getByName("45.22.30.39");
        InetAddress address2
            = InetAddress.getByName("45.22.30.39");
        InetAddress address3
            = InetAddress.getByName("172.19.25.29");

        // true, as clearly seen above
        System.out.println(
            "Is Address-1 equals to Address-2? : "
            + address1.equals(address2));
        // false
        System.out.println(
            "Is Address-1 equals to Address-3? : "
            + address1.equals(address3));

        // returns IP address
        System.out.println("IP Address : "
            + address1.getHostAddress());
        // returns host name,
        // which is same as IP
```

```

// address in this case
System.out.println(
    "Host Name for this IP Address : "
    + address1.getHostName());

// returns address in bytes
System.out.println("IP Address in bytes : "
    + address1.getAddress());

// false, as the given site
// has only one server
System.out.println("Is this Address Multicast? : "
    + address1.isMulticastAddress());

System.out.println("Address in string form : "
    + address1.toString());

// returns fully qualified
// domain name for this IP address.
System.out.println(
    "Fully qualified domain name for this IP address : "
    + address1.getCanonicalHostName());

// hashCode for this IP address.
System.out.println("HashCode for this IP address : "
    + address1.hashCode());

// to check if the InetAddress is
// an unpredictable address..
System.out.println(
    "Is the InetAddress an unpredictable address? : "
    + address1.isAnyLocalAddress());
}
}

```

### Output

```

Is Address-1 equals to Address-2? : true
Is Address-1 equals to Address-3? : false
IP Address : 45.22.30.39
Host Name for this IP Address : 45.22.30.39
IP Address in bytes : [B@579bb367
Is this Address Multicast? : false
Address in string form : 45.22.30.39/45.22.30.39

```



Fully qualified domain name for this IP address : 45.22.30.39

Hashcode for this IP address : 756424231

Is the InetAddress an unpredictable address? : false

**Video Content / Details of website for further learning (if any):**

- [java.net.InetAddress Class in Java - GeeksforGeeks](http://java.net.InetAddress Class in Java - GeeksforGeeks)
- <https://www.youtube.com/watch?v=4gpgYOTVSGA>
- <https://www.youtube.com/watch?v=Kx6i9gwNS3w>

**Important Books/Journals for further learning including the page nos.:**

- H.M. Deitel,P.J.Deitel,"Java How to Program" Prentice – Hall of India ,Newdelhi,
- Page No -1120

**Course Faculty**

**Verified by HOD**



# MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)  
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu



## LECTURE HANDOUTS

L - 44

MCA

I/II

Course Name with Code : 19CAB11 & Internet And Java Programming

Course Faculty : Mrs.G.Krishnaveni

Unit : JAVA BEANS AND NETWORKING

Date of Lecture: 19.04.2021

### Topic of Lecture: TCP/IP Client Sockets

#### Introduction : ( Maximum 5 sentences)

- There are two kinds of TCP sockets in Java. One is for servers, and the **other is for clients**.
- The `ServerSocket` class is designed to be a "listener," which waits for clients to connect before doing anything. Thus, `ServerSocket` is for servers

#### Prerequisite knowledge for Complete understanding and learning of Topic:

- Basic Networking and sockets

#### Detailed content of the Lecture:

- TCP/IP sockets are used to implement reliable, bidirectional, persistent, point-to-point, stream-based connections between hosts on the Internet.
- A socket can be used to connect Java's I/O system to other programs that may reside either on the local machine or on any other machine on the Internet.
- There are two kinds of TCP sockets in Java. One is for servers, and the other is for clients. The **ServerSocket** class is designed to be a "listener," which waits for clients to connect before doing anything.
- Thus, **ServerSocket** is for servers. The **Socket** class is for clients. It is designed to connect to server sockets and initiate protocol exchanges.
- Because client sockets are the most commonly used by Java applications, they are examined here.
- The creation of a **Socket** object implicitly establishes a connection between the client and server. There are no methods or constructors that explicitly expose the details of establishing that connection. Here are two constructors used to create client sockets:

Socket(String <i>hostName</i> , int <i>port</i> ) throws UnknownHostException, IOException	Creates a socket connected to the named host and port.
Socket(InetAddress <i>ipAddress</i> , int <i>port</i> ) throws IOException	Creates a socket using a preexisting <b>InetAddress</b> object and a port.

**Socket** defines several instance methods. For example, a **Socket** can be examined at any time for the address and port information associated with it, by use of the following methods:

InetAddress getInetAddress( ) : Returns the InetAddress associated with the Socket object. It returns null if the socket is not connected.

int getPort( ) : Returns the remote port to which the invoking Socket object is connected. It returns 0 if the socket is not connected.

int getLocalPort( ) : Returns the local port to which the invoking Socket object is bound. It returns -1 if the socket is not bound.

You can gain access to the input and output streams associated with a **Socket** by use of the **getInputStream( )** and **getOutputStream( )** methods, as shown here. Each can throw an **IOException** if the socket has been invalidated by a loss of connection. These streams are used exactly like the I/O streams described in Chapter 20 to send and receive data.

InputStream getInputStream( ) throws IOException : Returns the InputStream associated with the invoking socket.

OutputStream getOutputStream( ) throws IOException : Returns the OutputStream associated with the invoking socket.

Several other methods are available, including **connect( )**, which allows you to specify a new connection; **isConnected( )**, which returns true if the socket is connected to a server; **isBound( )**, which returns true if the socket is bound to an address; and **isClosed( )**, which returns true if the socket is closed. To close a socket, call **close( )**. Closing a socket also closes the I/O streams associated with the socket. Beginning with JDK 7, **Socket** also implements **AutoCloseable**, which means that you can use a **try-with-resources** block to manage a socket.

**The following program provides a simple Socket example.**

- It opens a connection to a "whois" port (port 43) on the InterNIC server, sends the command-line argument down the socket, and then prints the data that is returned. InterNIC will try to look up the argument as a registered Internet domain name, and then send back the IP address

and contact information for that site.

```
// Demonstrate Sockets.
```

```
import java.net.*; import java.io.*;
```

```
class Whois {
```

```
public static void main(String args[]) throws Exception { int c;
```

```
    Create a socket connected to internic.net, port 43. Socket s = new Socket("whois.internic.net", 43);
```

```
    Obtain input and output streams.
```

```
    InputStream in = s.getInputStream();
```

```
    OutputStream out = s.getOutputStream();
```

```
    // Construct a request string.
```

```
    String str = (args.length == 0 ? "MHProfessional.com" : args[0]) + "\n";
```

```
    // Convert to bytes.
```

```
    byte buf[] = str.getBytes();
```

```
    Send request. out.write(buf);
```

```
    //Read and display response.
```

```
    while ((c = in.read()) != -1) { System.out.print((char) c);
```

```
}
```

```
s.close();
```

```
}
```

```
}
```

If, for example, you obtained information about **MHPProfessional.com**, you'd get something similar to the following:

Whois Server Version 2.0

Domain names in the .com and .net domains can now be registered

with many different competing registrars. Go to <http://www.internic.net> for detailed information.

Domain Name: MHPROFESSIONAL.COM Registrar: CSC CORPORATE DOMAINS, INC. Whois Server: whois.corporatedomains.com Referral URL: <http://www.cscglobal.com> Name Server: NS1.MHEDU.COM

Name Server: NS2.MHEDU.COM

.

- Here is how the program works.
- First, a **Socket** is constructed that specifies the host name "whois.internic.net" and the port number 43. **Internic.net** is the InterNIC web site that handles whois requests.
- Port 43 is the whois port. Next, both input and output streams are opened on the socket. Then, a string is constructed that contains the name of the web site you want to obtain information about.
- In this case, if no web site is specified on the command line, then "MHPProfessional.com" is used. The string is converted into a **byte** array and then sent out of the socket.
- The response is read by inputting from the socket, and the results are displayed. Finally, the socket is closed, which also closes the I/O streams.
- In the preceding example, the socket was closed manually by calling **close( )**.
- If you are using JDK 7 or later, then you can use a **try-with-resources** block to automatically close the socket. For example, here is another way to write the **main( )** method of the previous program:

**// Use try-with-resources to close a socket.**

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

```
{
```

```
int c;
```

```
//Create a socket connected to internic.net, port 43. Manage this

//socket with a try-with-resources block.

try ( Socket s = new Socket("whois.internic.net", 43) ) {

    //Obtain input and output streams.
    InputStream in = s.getInputStream();
    OutputStream out = s.getOutputStream();

    Construct a request string.

    String str = (args.length == 0 ? "MHProfessional.com" : args[0]) + "\n"; // Convert to bytes.

    byte buf[] = str.getBytes();

    Send request. out.write(buf);

    //Read and display response.

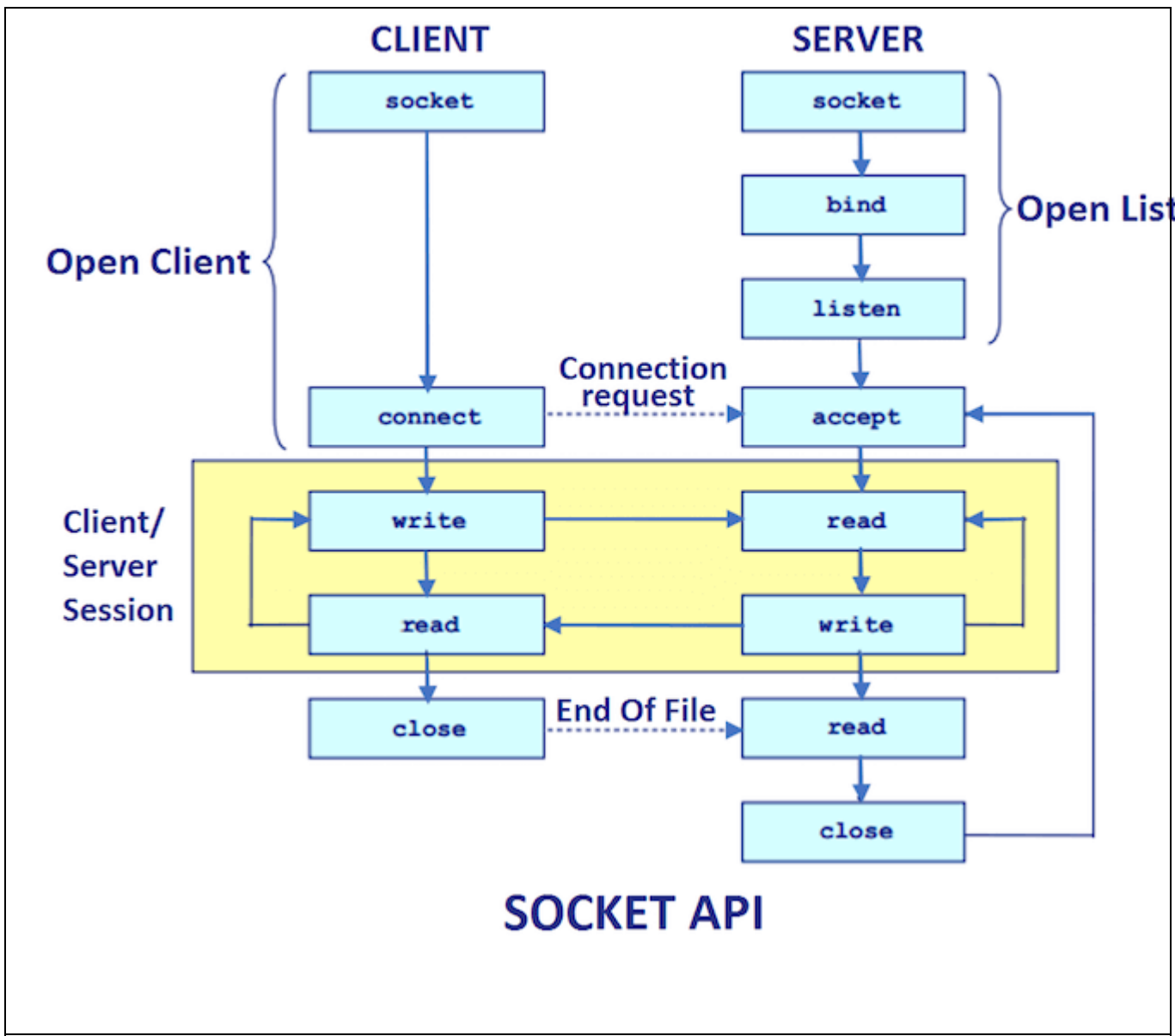
    while ((c = in.read()) != -1) { System.out.print((char) c);

}

}

// The socket is now closed.

}
```



## SOCKET API

**Video Content / Details of website for further learning (if any):**

- [TCP/IP Client Sockets - Java \(brainkart.com\)](http://brainkart.com)
- [Java Socket Programming \(Java Networking Tutorial\) - javatpoint](http://javatpoint.com)

**Important Books/Journals for further learning including the page nos.:**

- H.M. Deitel,P.J.Deitel,"Java How to Program" Prentice – Hall of India ,Newdelhi,
- Page No -1130

**Course Faculty**

**Verified by HOD**



# MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)  
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu



## LECTURE HANDOUTS

L - 45

MCA

I/II

Course Name with Code : 19CAB11 & Internet And Java Programming

Course Faculty : Mrs.G.Krishnaveni

Unit : JAVA BEANS AND NETWORKING

Date of Lecture: 20.04.2021

### Topic of Lecture: TCP/IP Server Sockets

#### Introduction : ( Maximum 5 sentences)

- Java Socket programming is used for communication between the applications running on different JRE.
- Java Socket programming can be connection-oriented or connection-less.

#### Prerequisite knowledge for Complete understanding and learning of Topic:

- Networking basic and Sockets

#### Detailed content of the Lecture:

- Java Socket programming is used for communication between the applications running on different JRE.
- Java Socket programming can be connection-oriented or connection-less.
- Socket and ServerSocket classes are used for connection-oriented socket programming and DatagramSocket and DatagramPacket classes are used for connection-less socket programming.

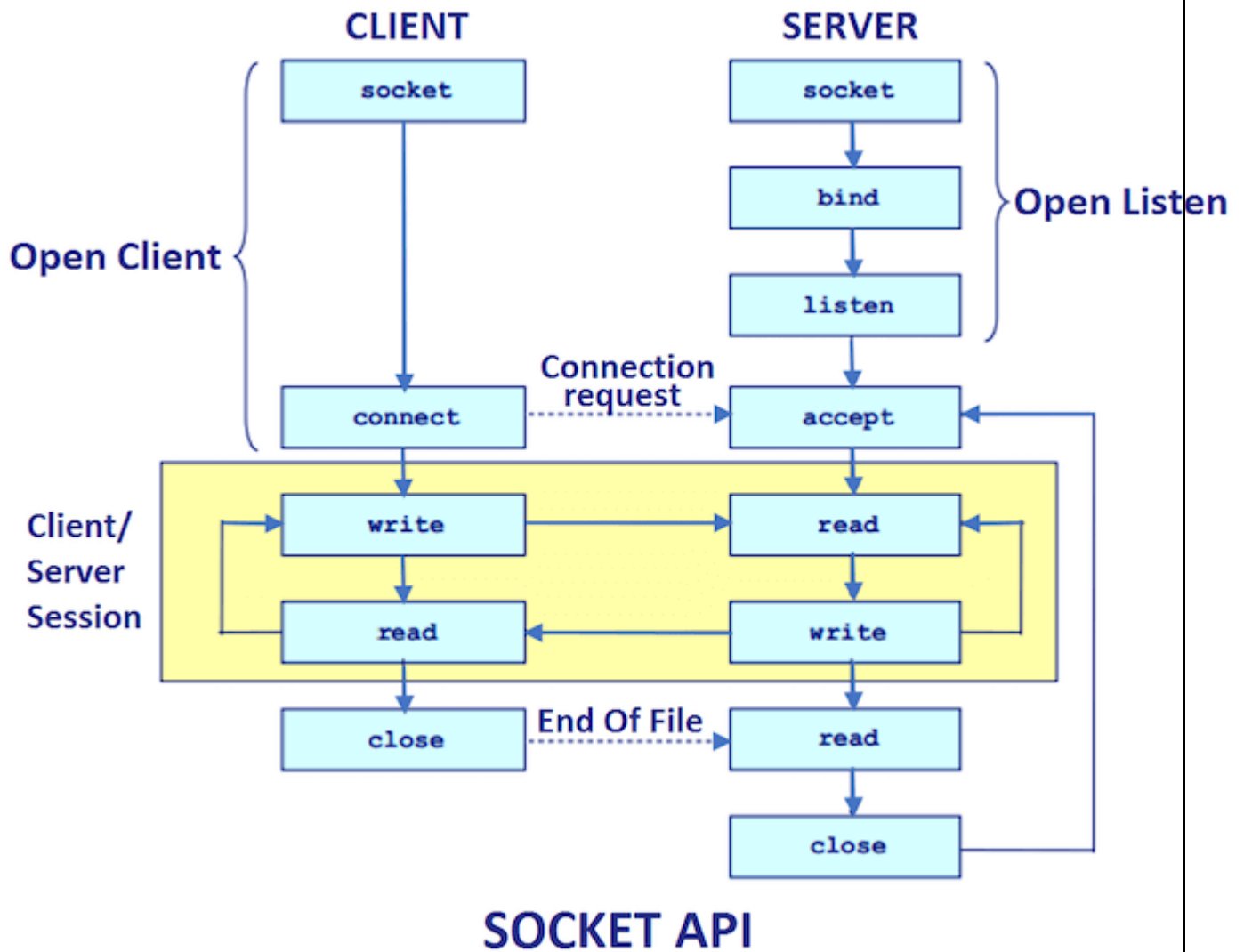
#### The client in socket programming must know two information:

1. IP Address of Server, and
2. Port number.

- Here, we are going to make one-way client and server communication.
- In this application, client sends a message to the server, server reads the message and prints it. Here, two classes are being used:
- Socket and ServerSocket. The Socket class is used to communicate client and server. Through this class, we can read and write message.
- The ServerSocket class is used at server-side.
- The accept() method of ServerSocket class blocks the console until the client is connected.



- After the successful connection of client, it returns the instance of Socket at server-side.



### Socket class

A socket is simply an endpoint for communications between the machines. The Socket class can be used to create a socket.

### Important methods

Method	Description
1) public InputStream getInputStream()	returns the InputStream attached with this socket.
2) public OutputStream getOutputStream()	returns the OutputStream attached with this socket.
3) public synchronized void close()	closes this socket

## ServerSocket class

The ServerSocket class can be used to create a server socket. This object is used to establish communication with the clients.

### Important methods

Method	Description
1) public Socket accept()	returns the socket and establish a connection between server and client.
2) public synchronized void close()	closes the server socket.

### Example of Java Socket Programming

#### Creating Server:

To create the server application, we need to create the instance of ServerSocket class. Here, we are using 6666 port number for the communication between the client and server. You may also choose any other port number. The accept() method waits for the client. If clients connects with the given port number, it returns an instance of Socket.

1. `ServerSocket ss=new ServerSocket(6666);`
2. `Socket s=ss.accept();//establishes connection and waits for the client`

#### Creating Client:

To create the client application, we need to create the instance of Socket class. Here, we need to pass the IP address or hostname of the Server and a port number. Here, we are using "localhost" because our server is running on same system.

1. `Socket s=new Socket("localhost",6666);`

Let's see a simple of Java socket programming where client sends a text and server receives and prints it.

*File: MyServer.java*

1. **import** java.io.\*;
2. **import** java.net.\*;
3. **public class** MyServer {
4. **public static void** main(String[] args){
5. **try**{
6. ServerSocket ss=**new** ServerSocket(6666);
7. Socket s=ss.accept();//establishes connection
8. DataInputStream dis=**new** DataInputStream(s.getInputStream());
9. String str=(String)dis.readUTF();
10. System.out.println("message= "+str);
11. ss.close();
12. }**catch**(Exception e){System.out.println(e);}
13. }
14. }

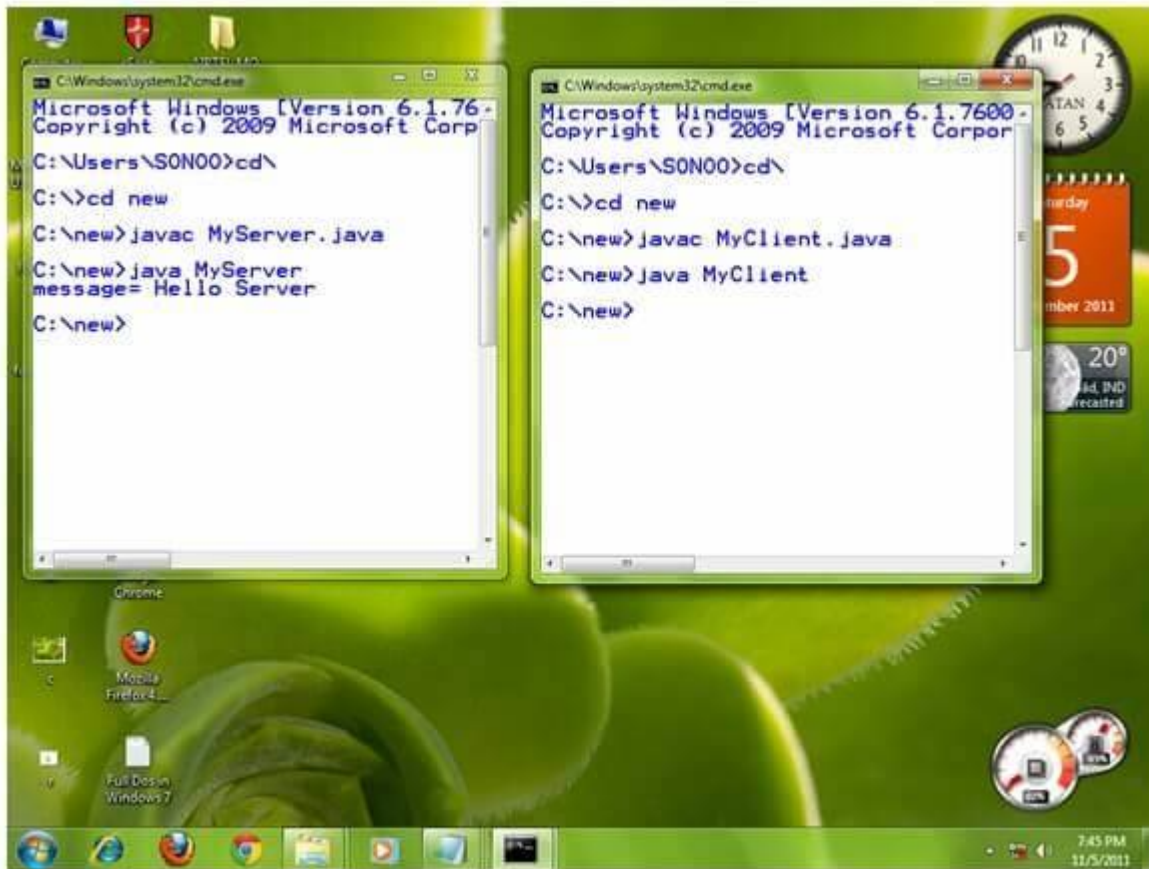
*File: MyClient.java*

1. **import** java.io.\*;
2. **import** java.net.\*;
3. **public class** MyClient {
4. **public static void** main(String[] args) {
5. **try**{
6. Socket s=**new** Socket("localhost",6666);
7. DataOutputStream dout=**new** DataOutputStream(s.getOutputStream());
8. dout.writeUTF("Hello Server");
9. dout.flush();
10. dout.close();
11. s.close();
12. }**catch**(Exception e){System.out.println(e);}
13. }
14. }

[download this example](#)

To execute this program open two command prompts and execute each program at each command prompt as displayed in the below figure.

After running the client application, a message will be displayed on the server console.



### Example of Java Socket Programming (Read-Write both side)

In this example, client will write first to the server then server will receive and print the text. Then server will write to the client and client will receive and print the text. The step goes on.

*File: MyServer.java*

1. **import** java.net.\*;
2. **import** java.io.\*;
3. **class** MyServer{
4. **public static void** main(String args[])**throws** Exception{
5. ServerSocket ss=**new** ServerSocket(3333);

```
6. Socket s=ss.accept();
7. DataInputStream din=new DataInputStream(s.getInputStream());
8. DataOutputStream dout=new DataOutputStream(s.getOutputStream());
9. BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
10.
11. String str="",str2="";
12. while(!str.equals("stop")){
13. str=din.readUTF();
14. System.out.println("client says: "+str);
15. str2=br.readLine();
16. dout.writeUTF(str2);
17. dout.flush();
18. }
19. din.close();
20. s.close();
21. ss.close();
22. }}
```

*File: MyClient.java*

```
1. import java.net.*;
2. import java.io.*;
3. class MyClient{
4. public static void main(String args[])throws Exception{
5. Socket s=new Socket("localhost",3333);
6. DataInputStream din=new DataInputStream(s.getInputStream());
7. DataOutputStream dout=new DataOutputStream(s.getOutputStream());
8. BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
9.
10. String str="",str2="";
11. while(!str.equals("stop")){
12. str=br.readLine();
13. dout.writeUTF(str);
14. dout.flush();
15. str2=din.readUTF();
```

```
16. System.out.println("Server says: "+str2);
17. }
18.
19. dout.close();
20. s.close();
21. }}
```

**Video Content / Details of website for further learning (if any):**

- [Java Socket Programming \(Java Networking Tutorial\) - javatpoint](#)
- <https://www.youtube.com/watch?v=BqBKEXLqdvI>

**Important Books/Journals for further learning including the page nos.:**

- H.M. Deitel,P.J.Deitel,"Java How to Program" Prentice – Hall of India ,Newdelhi,
- Page No -1135

**Course Faculty**

**Verified by HOD**