****

The Egyptian Ministry of Higher Education

Higher Technological Institute

10th of Ramadan City

Computer Science Department

**Questionnaire And Complains Management System For HTI Students**

**ANDROID & WEB APPLICATION**

**Graduation Project**

**CSC 400**

**Created by:**

Mohammed Reda Abdul Ghani 42015085

Nehal Gamal Mohamed 42016062

Belal Elsayed Elhalawaty 42015039

Ahmed Mohamed Sameh 42016071

Mahmoud Ahmed Elebiary 42015098

**Supervised by**

**Prof.DR.Eid Abd-El Hakim**

**June –August/ 201**9

**Acknowledgement**

I want to thank all who supported me in going in this graduation

project and where source of inspiration for me in gaining more experience. Also, I would like to express my deep gratitude to all my professors in HTI, who guided me through this Project.

I respect the work which done every day by engineers in our department thanks for them all.

Also I want to thanks **Dr.Eid Abd El Hakeim** to provide us with a good information and guide us to the right way.

I want to thank all those who supported me in this training.

**Prof.Dr.Eid Abd El Hakeim** , **Dr.Rania Ragab** and **Dr.Hosny Al -Mutafi** fortheir great supervising efforts to make me gain the mostexperience and for efforts made by me to make me understand the meaning of this graduation project.

**CHAPTER ONE**

PROBLEM DEFINITION AND

SYSTEM ANALYSIS

# 1.1 INTRODUCTION

## 1.1.1What is Questionnaire & Complain system?

Questionnaire system is made to help students and Institute Management to easy the questionnaire process.

When students download the mobile application or browsing the website and sign in they can done it's all subjects questionnaire by easy way and the result will not appear to students if they hadn't done its subjects

In case of institute management & system admen  can access the questionnaire result to every subject with every doctor in easy way because the result will generate automatically when the student completed There questionnaire

Complain system is made to help students and Institute Management to be always in touch and the students can deliver their demands and problems to the management and management can solve it

# 1.2 OBJECTIVE

## 1.2.1 definition for Questionnaire & Complain system Problem

Students find difficulty to complete a questionnaire paper for each subject and there is some students are not complete the questionnaire and some students complete the questionnaire but by wrong manner and their questionnaire are invalid.

Questionnaire administrators make a hard effort to scan the questionnaire paper to all  students in the institute and get results to every doctor on every department in institute

Students face some problems during their years of study at the Institute, and don't know who is responsible person for receive their complains

## 1.2.2 Suggested Solutions

Our team develop a computerized system (web application-mobile application)  to convert all this paperwork to automatic process done when students done their questionnaires easily. the result will appear on admin panel screen by charts and numbers. And complains of student will appear first to Head of Department In order to solve problems that are within his powers and then the rest of the unsolved problems will go to the management.

## 1.2.3 Goal of the project

The application aimed to facilitate the process of the questionnaire & complain by access the application at any time anywhere and done the questionnaires easily and automatically the result will be calculated. and also application aimed to establish Connection Channel between students and institute management.

# 1.3 SYSTEM ANALYSIS

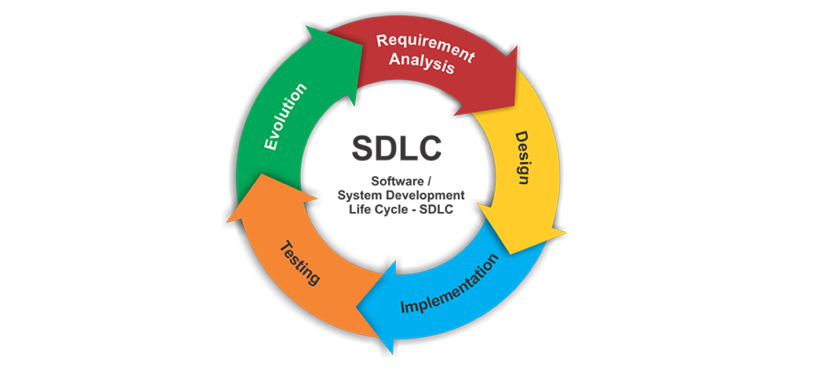
## 1.3.1 What is the system analysis?

System analysis is the body of interaction entities including computer system analysis. This field is closely related to requirement analysis operation research.

A person who analyze system is known as analyst. Often systems analysts employed by organization of businesses to help them improve their system and so became more efficient and for businesses, more profitable

## 1.3.2 The systems development life cycle (SDLC)

The software development life cycle (SDLC) is a framework defining tasks performed at each step in the software development process. SDLC is a structure followed by a development team within the software organization. It consists of a detailed plan describing how to develop, maintain and replace specific software



**Figure 1.1:** SDLC life cycle

## 1.3.3 Waterfall Model

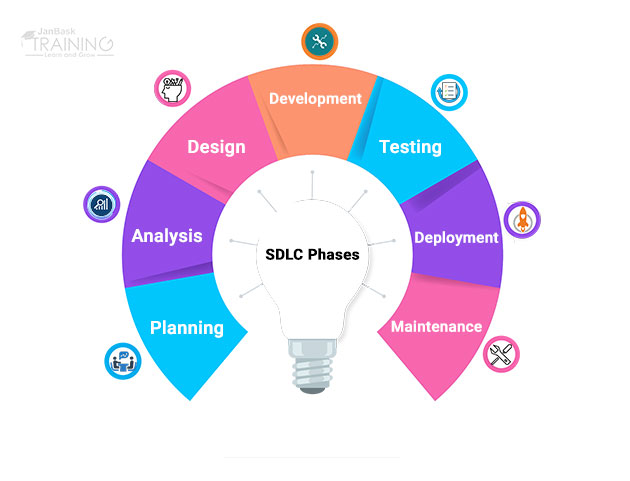
The Waterfall Model was first Process Model to be introduced. It is very simple to understand and use. In a Waterfall model, each phase must be completed before the next phase can begin and there is no overlapping in the phases. Waterfall model is the earliest SDLC approach that was used for software development. In “The Waterfall” approach, the whole process of software development is divided into separate phases. The outcome of one phase acts as the input for the next phase sequentially. This means that any phase in the development process begins only if the previous phase is complete. The waterfall model is a sequential design process in which progress is seen as flowing steadily downwards (like a waterfall) through the phases of Conception, Initiation, Analysis, Design, Construction, Testing, Production/Implementation and Maintenance. As the Waterfall Model illustrates the software development process in a linear sequential flow; hence it is also referred to as a Linear-Sequential Life Cycle Model.

-**Sequential Phases in Waterfall Model**

* **Requirements:**The first phase involves understanding what need to be design and what is its function, purpose etc. Here, the specifications of the input and output or the final product are studied and marked.
* **System Design:** The requirement specifications from first phase are studied in this phase and system design is prepared. System Design helps in specifying hardware and system requirements and also helps in defining overall system architecture. The software code to be written in the next stage is created now.
* **Implementation:** With inputs from system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality which is referred to as Unit Testing.
* **Integration and Testing:** All the units developed in the implementation phase are integrated into a system after testing of each unit. The software designed, needs to go through constant software testing to find out if there are any flaw or errors. Testing is done so that the client does not face any problem during the installation of the software.
* **Deployment of System:** Once the functional and non-functional testing is done, the product is deployed in the customer environment or released into the market.
* **Maintenance:** This step occurs after installation, and involves making modifications to the system or an individual component to alter attributes or improve performance. These modifications arise either due to change requests initiated by the customer, or defects uncovered during live use of the system. Client is provided with regular maintenance and support for the developed software.

All these phases are cascaded to each other in which progress is seen as flowing steadily downwards (like a waterfall) through the phases. The next phase is started only after the defined set of

goals are achieved for previous phase and it is signed off



**Figure 1.2:** Water Full Model

**1.3.4 Our Architecture pattern ( MVC )**

* **The Model-View-Controller (MVC)** :

is an architectural pattern that separates an application into three main logical components: the model, the view, and the controller. Each of these components are built to handle specific development aspects of an application. MVC is one of the most frequently used industry standard web development framework to create scalable and extensible projects.

* **Model:**

The Model component corresponds to all the data-related logic that the user works with. This can represent either the data that is being transferred between the View and Controller components or any other business logic-related data. For example, a Customer object will retrieve the customer information from the database, manipulate it and update it data back to the database or use it to render data.

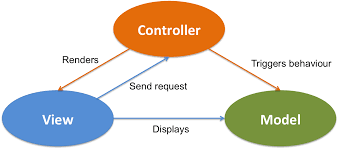
* **View**:

The View component is used for all the UI logic of the application. For example, the Customer view will include all the UI components such as text boxes, dropdowns, etc. that the final user interacts with

* **Controller:**

Controllers act as an interface between Model and View components to process all the business logic and incoming requests, manipulate data using the Model component and interact with the Views to render the final output. For example, the Customer controller will handle all the interactions and inputs from the Customer View and update the database using the Customer Model. The same controller will be used to view the Customer data

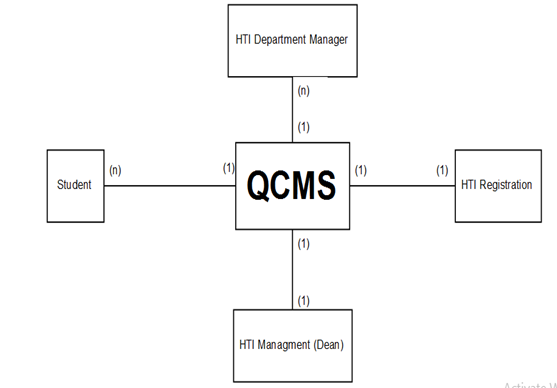
**MVC Components**: Following are the components of MVC



**Figure 1.3:** MVC Model

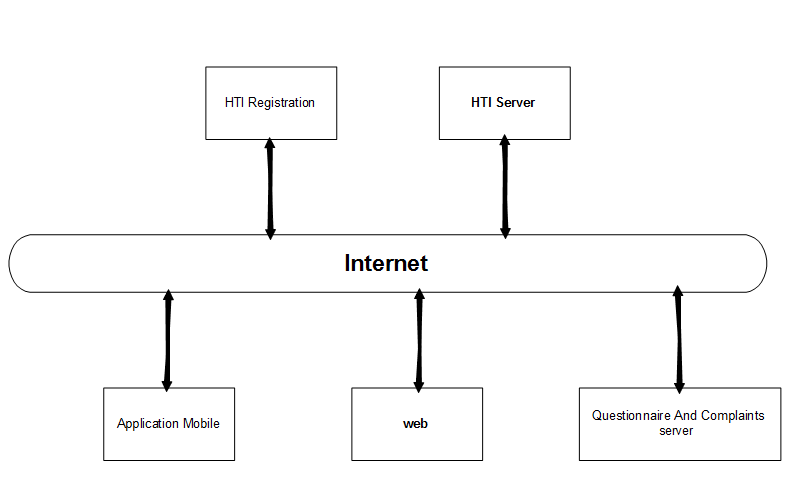
## 1.3.5 Context Diagram

Context model are used to illustrate the operational context of a system. A system context model that demonstrates the other system in the environment of the system being developed



**Figure 1.4:** CONTEXT Model

## 1.3.6 Architecture Diagram



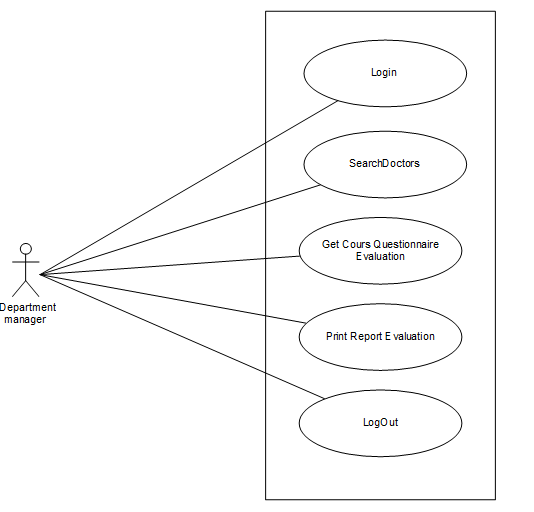
**Figure 1.5:** CONTEXT Model

## 1.3.7 Use Case (UML)

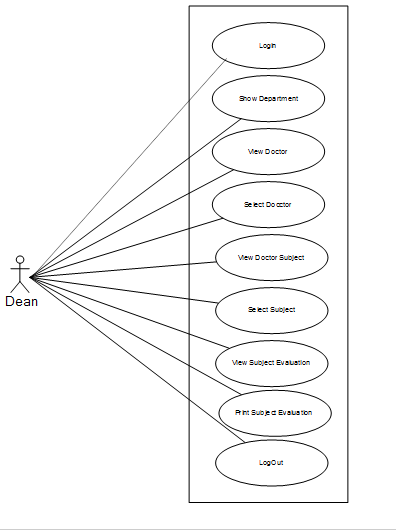
A use case is a methodology used in system analysis to identify, clarify, and organize system requirements. The use case is made up of a set of possible sequences of interactions between systems and users in a particular environment and related to a particular goal.

****

**Figure 1.6:** USE CASE Model



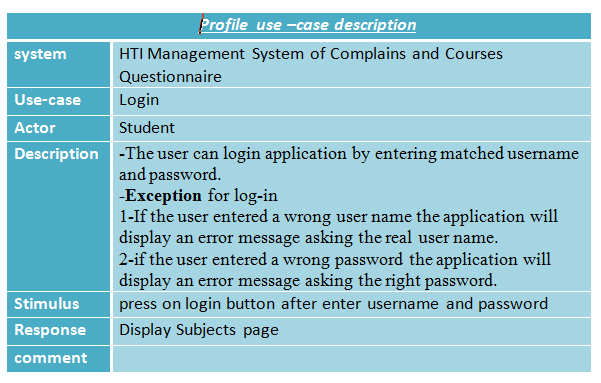
**Figure 1.7:** USE CASE Model

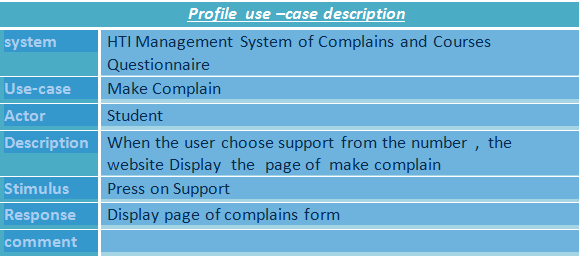
****

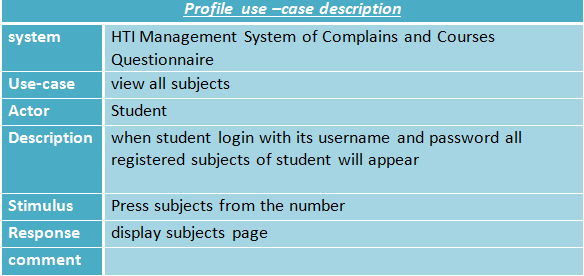
**Figure 1.8:** USE CASE Model

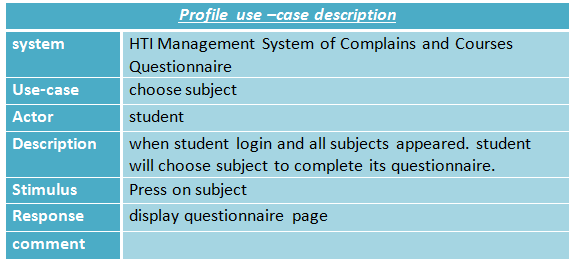
## 1.3.8 Use-case scenario:

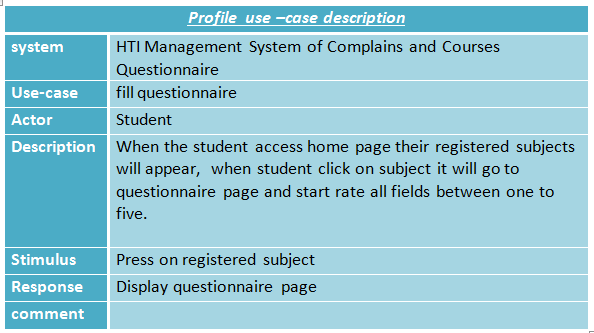
student scenarios:

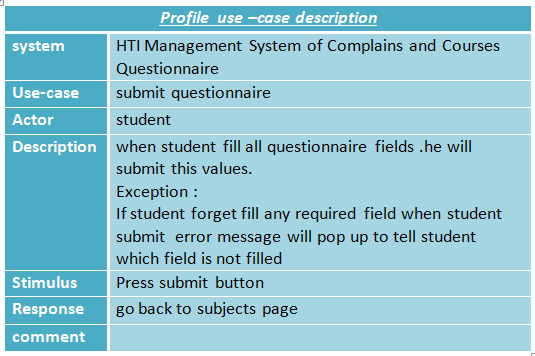
****

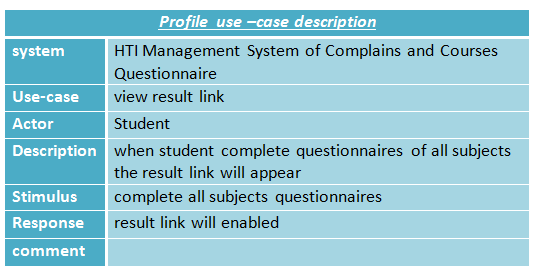
****

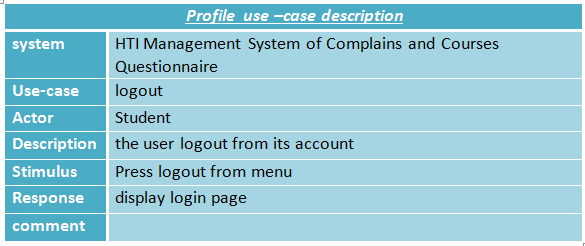
****

****

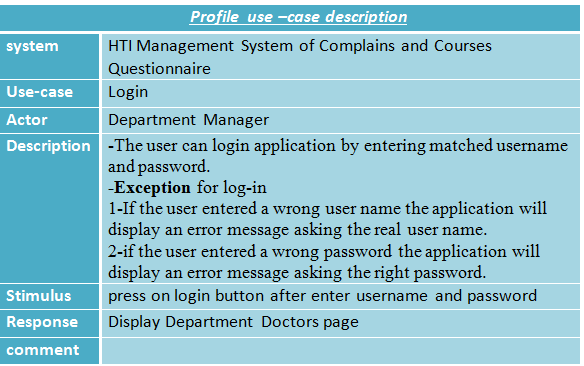
****

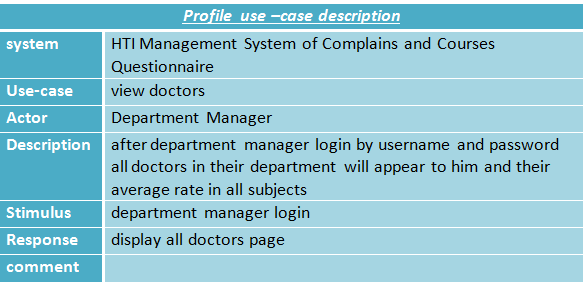
****

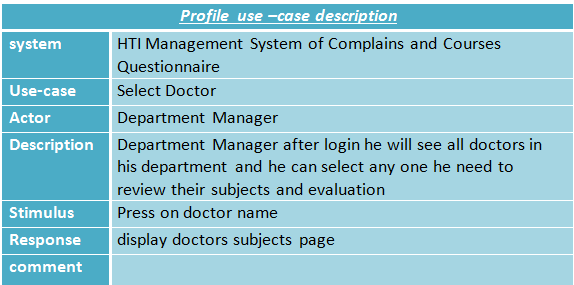
****

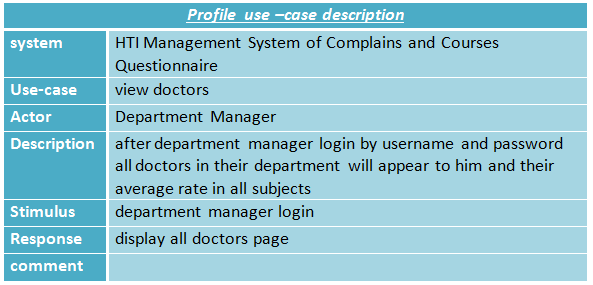
****

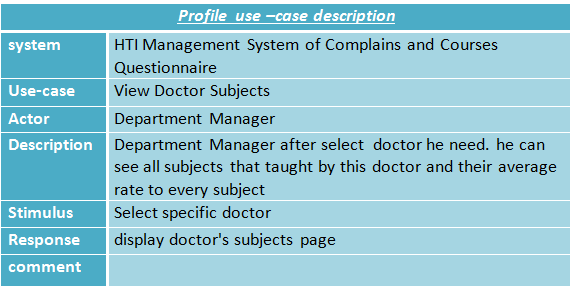
Department manager scenario:

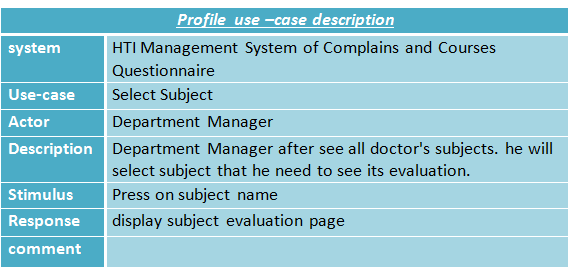


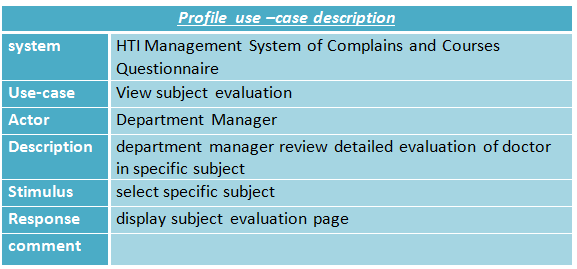


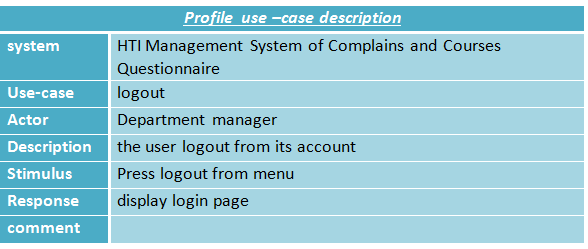
****

****

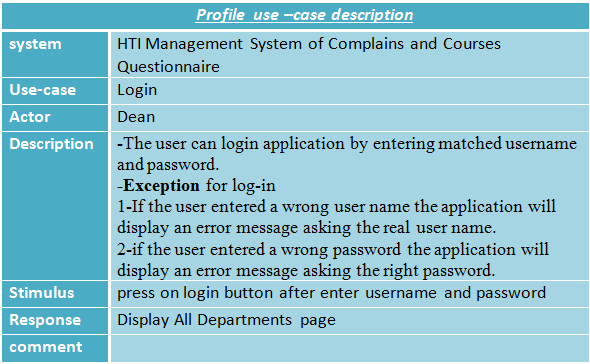
****

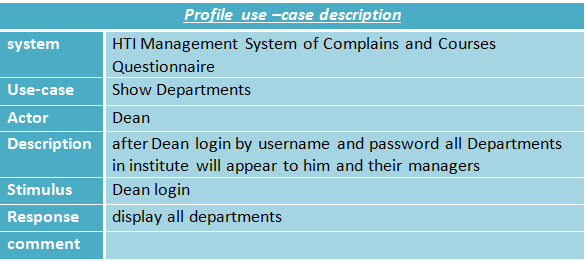
****

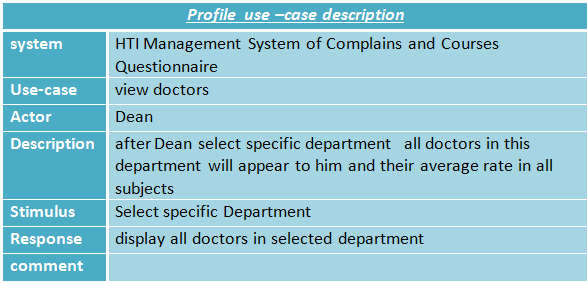
****

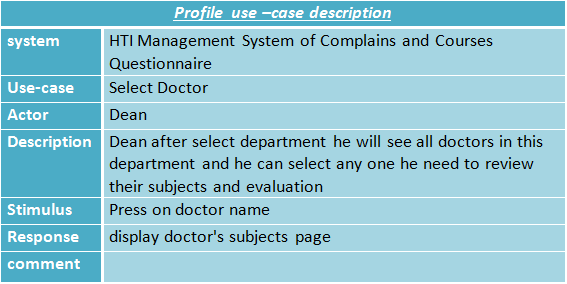
****

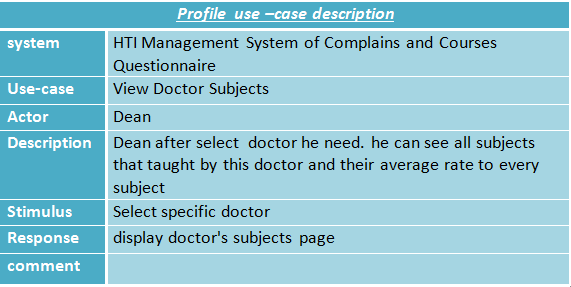
Dean scenario:

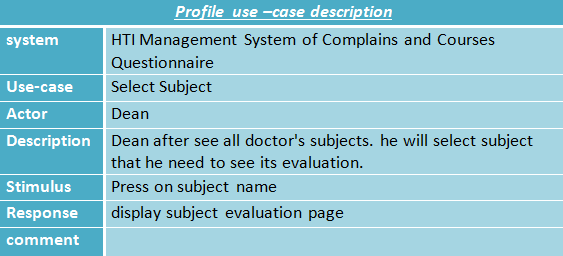


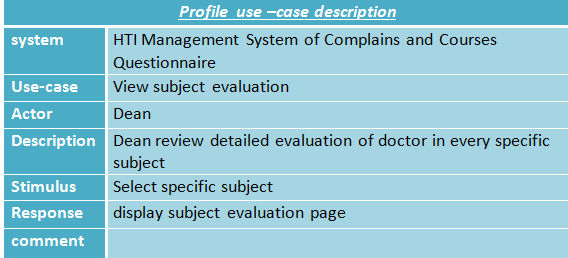
****

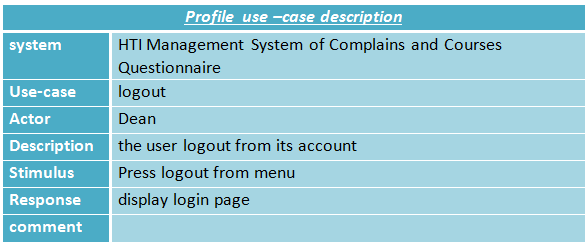
****

****

****

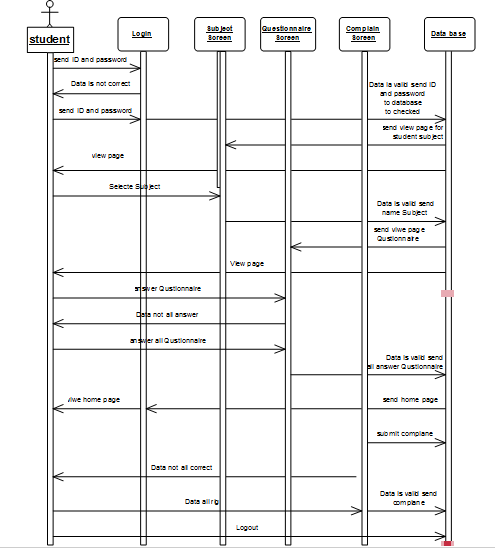
****

****

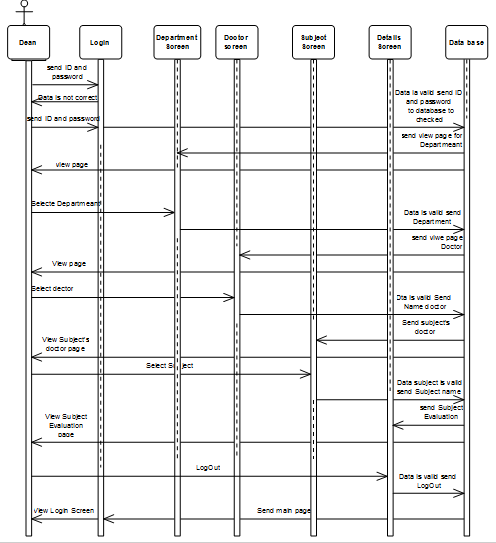
****

## 1.3.9 Sequence Diagram

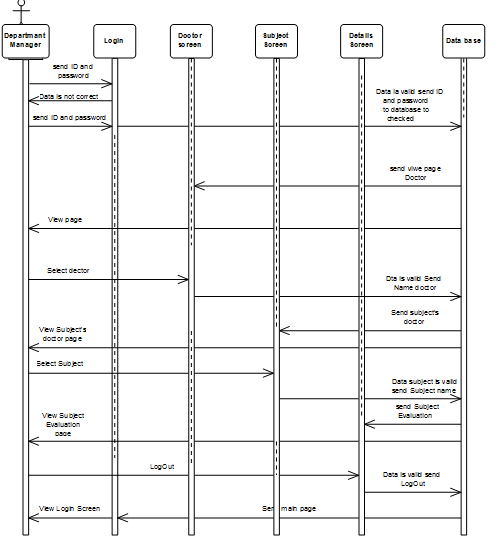
UML Sequence Diagrams are interaction diagrams that detail how operations are carried out. They capture the interaction between objects in the context of a collaboration. Sequence Diagrams are time focus and they show the order of the interaction visually by using the vertical axis of the diagram to represent time what messages are sent and when.



**Figure 1.9:**Sequence DiagramModel



**Figure 1.10:**Sequence DiagramModel



**Figure 1.11:**Sequence DiagramModel

## 1.3.10 ER Diagram

An entity relationship diagram (ERD) shows the relationships of entity sets stored in a database. An entity in this context is an object, a component of data. An entity set is a collection of similar entities. These entities can have attributes that define its properties By defining the entities, their attributes, and showing the relationships between them, an ER diagram illustrates the logical structure of databases ER diagrams are used to sketch out the design of a database.

**Figure 1.12:** ER Diagram model

## 1.3.11 Dialogue diagramming :

A method for designing and representing dialogues is dialogue diagramming**.**

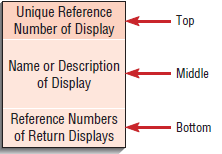
Dialogue diagrams have only one symbol, a box with three sections; each box represents one display (which might be a full screen or

a specific form or window) within a dialogue. The three sections of the box are used as follows :

1. Top: Contains a unique display reference number used by other displays for referencing it

2. Middle: Contains the name or description of the display

3. Bottom: Contains display reference numbers that can be accessed from the current display



**Figure 1.13:** dialogue-diagramming box has three sections



**Figure 1.14:**Dialogue diagrammingModel

**CHAPTER TWO**

(WEB APPLICATION)

# 2.1   WHAT'S WEB DESIGN!?

**Design is**

the process of collecting ideas, and aesthetically arranging and implementing them, guided by certain principles for a specific purpose. Web design is a similar process of creation, with the intention of presenting the content on electronic web pages, which the end-users can access through the internet with the help of a web browser.

## 2.1.1 Elements of Web Design

Web design uses many of the same key visual elements as all types of design such as:

**Layout:**This is the way the graphics, ads and text are arranged. In the web world, a key goal is to help the view find the information they seek at a glance. This includes maintaining the balance, consistency, and integrity of the design.

**Color:** The choice of colors depends on the purpose and clientele; it could be simple black-and-white to multi-colored design, conveying the personality of a person or the brand of an organization, using web-safe colors.

**Graphics:** Graphics can include logos, photos, clipart or icons, all of which enhance the web design. For user friendliness, these need to be placed appropriately, working with the color and content of the web page, while not making it too congested or slow to load.

**Fonts:**The use of various fonts can enhance a website design. Most web browsers can only read a select number of fonts, known as "web-safe fonts", so your designer will generally work within this widely accepted group.

**Content:**Content and design can work together to enhance the message of the site through visuals and text. Written text should always be relevant and useful, so as not to confuse the reader and to give them what they want so they will remain on the site. Content should be optimized for search engines and be of a suitable length, incorporating relevant keywords

.

## 2.1.2 Technologies of Web Design

* + HTML
  + CSS
  + JavaScript
  + JQuery
  + Bootstrap

# 2.2   OVERVIWE OF HTML

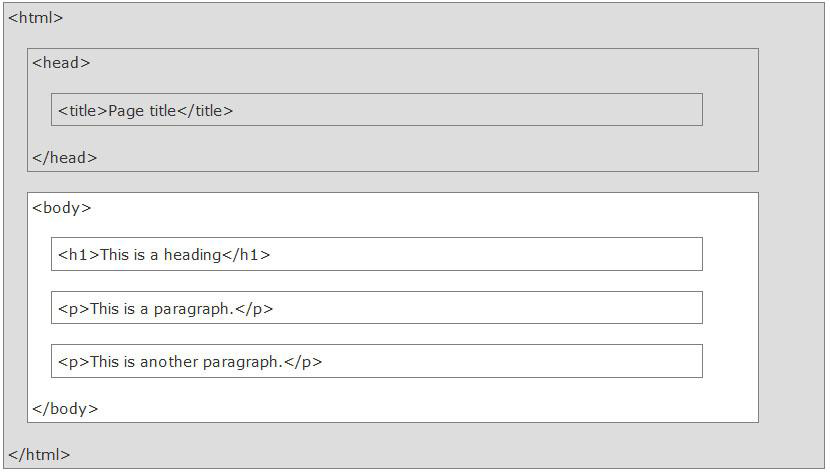
## 2.2.1 What is the HTML?

HTML is an acronym for Hyper Text Markup Language. It is a static language that determines the structure and semantic meaning of a web page. You use HTML to create content and metadata that browsers use to render and display information. HTML content can include text, images, audio, video, forms, lists, tables, and many other items. An HTML page can also contain hyperlinks, which connect pages to each other and to websites and resources elsewhere on the Internet.

## 2.2.2 Structure of HTML Page

Every HTML page has the same basic structure:

* + A DOCTYPE declaration stating which version of HTML the page uses.
  + An html section that contains the following elements:
  + A header that co
  + ntains information about the page for the
  + browser. This may include its primary language (English,
  + Chinese, French, and so on), character set, associated style sheets and script files, author information, and keywords for search engines.
  + A body that contains all the viewable content of the page. This is true for all versions of HTML up to and including HTML5. An HTML5 web page should include a DOCTYPE declaration, and a <html> element that in turn contains a <head> element containing the title and character set of the page and a <body> element for the content.



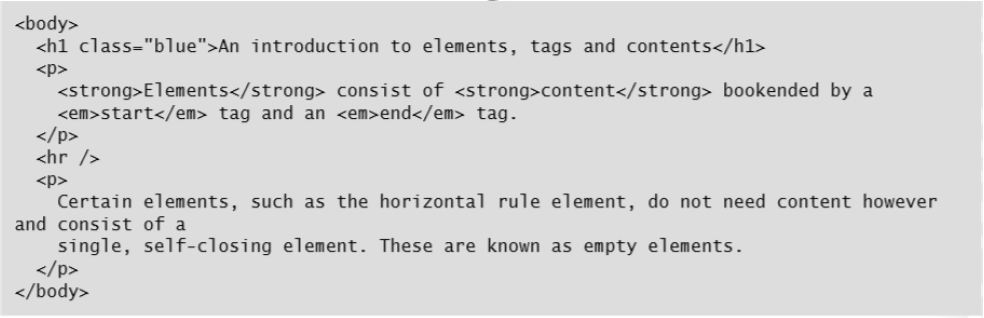
## 2.2.3 Tags, Elements, Attributes of HTML

The head and body of a web page both use HTML elements to define its

structure and Contents.

For example, a paragraph element, representing a paragraph of text on the page, consists of:

* An opening tag, <p>: to denote the start of the paragraph.
* Text content.
* A closing tag, </p>: to denote the end of the paragraph



Each HTML element tells the browser something about the information that sits between its opening and closing tags.

For example, the strong and **em** elements represent **“strong importance"** and

**“emphasis"** for their contents, which browsers tend to render respectively as text in bold andtext in italics. **H1** elements represent a top-level heading in your document, which browsers tend to render as text in a large, bold font.

Attributes provide additional information, presentational or semantic, about the contents of an element. They appear within the start tag of an element and have a name and a value.

The name should be in lowercase characters. Most values are pre-defined based on the attribute they are for, and should be contained within quotes.

In the previous example, the h1 start tag contains the class attribute set to the value blue. Most attributes can qualify only certain elements. However, HTML defines a group of global attributes for use with any element.

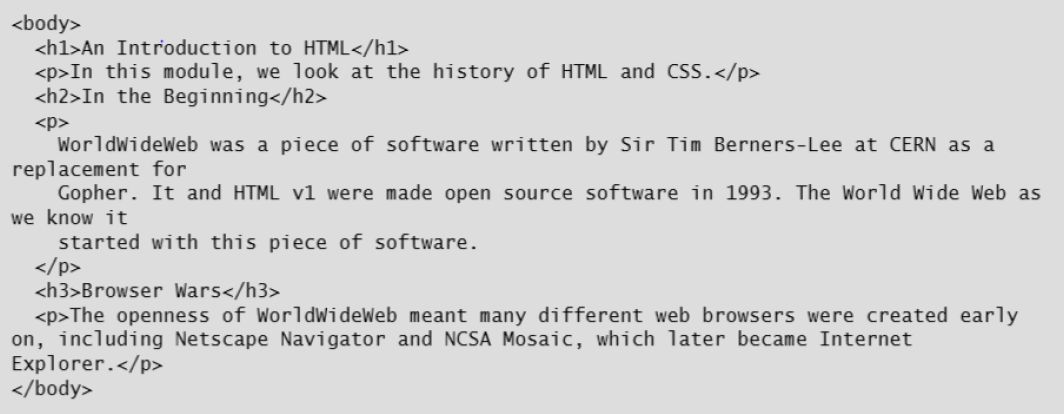
## 2.2.4 Heading and Paragraphs

HTML has included elements to identify paragraphs and headings in a document since v1 in 1992.

<p> identifies paragraphs of text.

<h1>, <h2>, <h3>… <h6> identify six levels of heading text.

Use <h1> to identify the main heading of the entire page, <h2> to identify the headings of each section in the page, <h3> to identify the sub-sections within those secondary headings, and so on



## 

## 2.2.5 Displaying Images in HTML

You use the HTML <img> tag to insert an image into your web page.

It does not require an end tag as it does not contain any content.

In addition to the global attributes, the <img> tag has a number of attributes to define it:

* + - The src attribute specifies a URL that identifies the location of the image to be displayed.
    - The alt attribute identifies a text alternative for display in place of browser is still downloading it or cannot display it for some Reason; for example, if the image file is missing.
    - It typically describes the content of the image.
    - The title attribute identifies some text to be used in a tool tip when a user’s cursor hovers over the image.
    - The height and width attributes set the dimensions in pixels of the box on the web page that will contain the image; if the dimensions are different from those of the image, browsers will resize the image on the fly to fit the box.

# 2.3  OVERVIEW OF CSS

## 2.3.1 CSS Syntax

CSS is an acronym for Cascading Style Sheets.

CSS provides the standard way of defining how a browser should display the contents of a web page.

CSS enables you to attach presentation rules to fragments of HTML based on selectors that target HTML elements by name, id, or class. It also enables you to vary how a page is presented according to the form factor of the device on which it is displayed, from a large monitor to a smart phone, and even on an audio reader or printer.

**Every CSS rule has the same basic structure:**

**This example shows the four parts of every CSS rule**:

A selector defines the element or set of elements to target.

The styling specified by the CSS rules associated with this selector is applied to all elements on the web page that match this selector.

A CSS selector can specify the type of element such as div to select all <div> elements, or the name attribute of a specific instance of an element. You can also select multiple element types such as p, div to select all <p>

and all <div>elements. You can even \* to select all elements. Other selector expressions are also possible, and these are described

later in this course.

* A pair of curly braces encloses the rules for the selected elements. A rule defines how to render the selected element; it contains a property-value pair suffixed by a semi-colon.
* A property identifies the visual aspect of the selected element to change.
* A values variable specifies the styling to apply to the property. Values can vary depending on the property. They might be color names, size values in percentages, pixels, ems, or points, or the name of a font, to name three possibilities.You can also add comments to your style sheets by using /\* \*/ delimiters. The browser.

## 

**FIG 2.5 CSS Rules**

## 2.3.2 How CSS Selectors Work

CSS selectors specify the content to be styled by using the associated set of rules, and understanding how CSS selectors work is the key to defining reusable and extensible style sheets.

The CSS specification provides many different ways to select the element or set of elements in a web page to which presentation rules will apply. The following list summarizes the basic selectors and the set of elements that they identify.

* The element selector identifies the group of all elements in the page with that name. For example, h2 {} returns the set of all level two headings in the page.
* The class selector, identified by a period, returns the set of all elements in the page where the class attribute is set to the specified value. For example, .myClass {} returns the set of elements where the class attribute is set to "myClass".
* The id selector, identified by a hash, returns the set of all elements in the page where the id attribute is set to the specified value. For example, #thisId {} returns the set of any elements where the id attribute is set to "thisId".

The class selector may return a set containing different types of HTML elements—for example, <p>, <section>, and <h3>—if they all have the same class attribute value. The same is true of the id selector, although this selector should only return a single element because the id attribute in a page should be unique (this is not enforced).

Style sheets are often written with the least specific selectors first and the most specific selectors last. Element selectors are the least specific, followed by class and id selectors, and then combinations of the three.



**FIG 2.6 Introducing the element, class, and id selectors**

You can combine selectors by using concatenation.

In the following example, the two rules combine selectors to identify a more specific set of elements than either does on its own.

The selector, h2.blue returns the set of <h2> elements with the class "blue", and h2#toc returns the set of <h2> elements with id "toc".

Note that these two sets may intersect, in which case the CSS properties and values for both rules will apply—in this case to the set of <h2> elements with id "toc" and class "blue".

The following table shows examples of the various ways you can concatenate

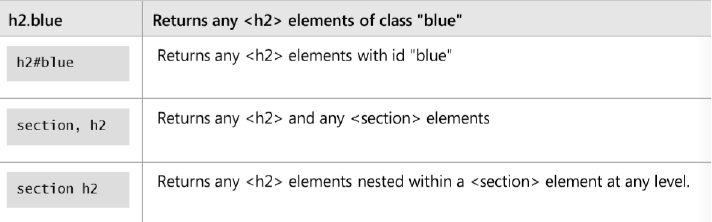
selectors and the set of elements that the browser return****Z

FIG 2.6 CSS Inheritance classes and ids

## 

## 2.3.3 Styling an HTML page using CSS

However good the layout of a website, it can be nullified if you write poor content or present it badly. If content is the foundation of the web, then typography is the foundation of web design, and so the first set of CSS properties that you should learn are those that concern the ways in which

text is displayed

C:\Users\m_sameh\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Screenshot (4).png

FIG 2.7 Fonts

The most obvious aspect of text content is the font. The discussion of when to use which type of font for headings, body text, sidebars, and so on is outside the scope of this course. Also, many organizations have style guides that specify the fonts for their web pages. That said, you can use the following CSS properties to set the selected font.

The following code shows some examples:

* **The font-size property,** which sets the height of the font. You can set the font-size value to an absolute value in pixels (for on-screen display), in points (for printing), or to a value relative to the parent

element’s font-size (in percent), or relative to the base font size of the page (in ems).

* **The font-style property**, which enables you to select a normal (vertical), italic, or oblique version of the font to be displayed.
* **The font-weight property**, which enables you to set the weight of a font. Usually, this means setting the property value to bold, but you can also set it to one of nine numerical values (100,200, … , 900)

reflecting different font weights in that family (black, book, semi-bold, and so on).

CSS also provides a shortcut property simply called font, which enables you to set some or all of these four properties (plus line-height) in a single rule rather than having to write out all five rules for every element.

You must set the value for these properties in the following order (note that the font-family and font-size properties are mandatory, but the other properties are optional):

1. Font-style.
2. Font-weight.
3. Font-size/line-height.
4. Font-family.

**Color**

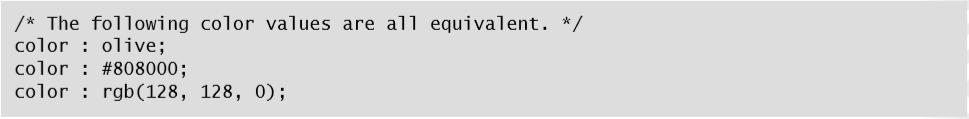
There are two color-related properties in CSS: color and opacity.

The color property, which enables you to set the color of a font.

You can specify the color as an RGB (red-green-blue) value or as one of the 147 predefined color names in the HTML and CSS specification. The following code shows some examples:

The opacity property, which enables you to set the transparency of some text or of an image. This property takes a value between 0.0 (fully transparent) and 1.0 (fully opaque). The following code shows an example:

FIG 2.9 Color Style



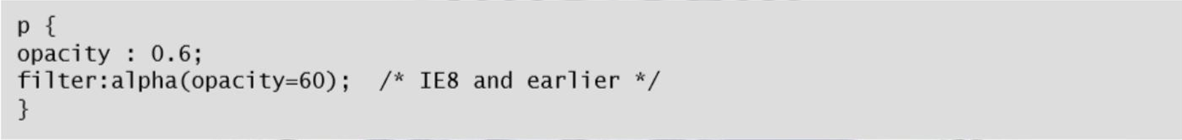
Note that Internet Explorer versions prior to Internet Explorer 9 do not support the opacity property. Instead, you must use the filter property and set the opacity to a value between 0 and 100.

FIG 2.10 Opacity Style

## 2.3.4 CSS Box

To determine the layout of an HTML page, browsers treat each element in the page as a set of four nested boxes. The CSS box model enables you to specify the size of each box, and so modify the layout of each element on the page. The box model places content inside four boxes: Content, Padding, Border, and Margin

In the center box is the content, your text and images. Use the height and width properties to set the height and width of the content box in pixels.

Around the content box is the padding box. Use the padding property to set the width of the padding box.

Around the padding box is the border box, which can also act as a visible line around the content and padding. Use the border property to set its width, color, and style.

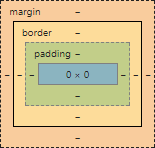
Around the border box is the margin box. Use the margin property to set the

FIG 2.11 CSS Box Sizing

width of the margin box.

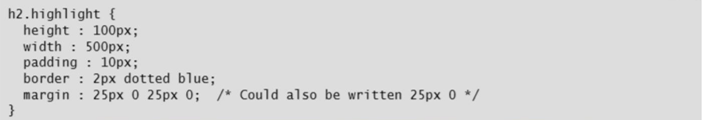
The following code example shows how to use the CSS box model to draw a border around some padded heading text and to set a margin around the border so that it does not interfere with other elements on the page.

FIG 2.12 Using box model properties

Border is also a shorthand property for the width, style, and color of the border box. In the previous example, border is set to 2px dotted blue. You could write this out in full as:

****

FIG 2.13 border

Using the border-width, border-style, and border-color properties assumes that you want the set values to be the same around all four sides of the box. If this is not the case or you only want to set them for one side of the border, you can use the border-left-style, border-left-width, border-left-color, border-right-style, border-right-width and so on properties. For example:

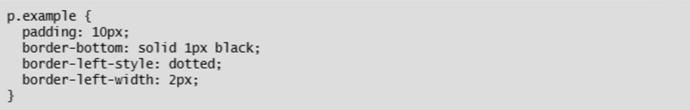
****

FIG 2.14 border

The border-top, border-right, border-bottom, and border-left properties are also shorthand properties, like border, but for the width, style, and color of the respective sides of the border box.

## 

## 2.3.5 Styling Backgrounds in CS

## Many websites use a background image, color, or pattern to provide more color and character to the pages. CSS enables you to set a background for any block-level element by using the following elements.

* **The background-image property**

which enables you to specify the URL of an image to use as a background for the selected elements.You may use a relative or an absolute URL. Note that if you provide a relative URL, you must specify the path relative to the location of the style sheet or web page that defines the style.

* **The background-size property**

, which enables you to set the height and width of the background image. Use values specified in pixels or as percentages of the height and width of the specified element.

* **The background-color property**

, which enables you to set the color of an element's background. You can specify the color as an RGB (red-green-blue) value or as one of the 147 predefined color names in the HTML and CSS specification.

* **The background-position property**

, which enables you to set the position of the background image in the element. The property takes two values: the first for the x-axis and the

second for the y-axis. Set them both as absolute values (top, left, bottom, right, center), percentages, or pixels

* **The background-repeat property**,

which enables you to set how a background image is repeated behind the selected element if it is smaller than the selected element. Possible values are repeat (the default), repeat-x, and repeat-y and no-repeat.

CSS also provides the background shortcut property, which enables you to set some or all of the elements just described. You must set the values for these properties in the following order (only the background-image property is mandatory, the others are optional):

1. Background-color.
2. Background-position.
3. Background-size.
4. Background-repeat.
5. Background-origin.
6. Background-clip.
7. Background-attachment.
8. Background-image.

# 2.4 OVERVIEW OF JAVA SCRIPT

## 2.4.1 What is the JavaScript!?

JavaScript originated as a programming language in the 1990s and has steadily evolved. The standard upon which it is based, ECMA-262, is frequently updated and augmented.

Consequently, the JavaScript engines used in modern browsers are now exponentially faster and more functional than their predecessors. At its heart, JavaScript is a scripting engine with the same basic features as any other programming language.

It provides:

• Variables for storing information.

• Operators for performing calculations and comparisons.

• Functions for grouping statements into reusable chunks.

• Conditional statements and loop constructs to control program flow.

• The ability to create objects with properties, methods, and events. By itself, JavaScript cannot do much more than perform calculations and

manipulate text, but in combination with the DOM that all browsers implement you can do far more.

For example, you can use JavaScript code to:

• Add or remove items to a list displayed on a page.

• Add, change, or remove text on a page.

• Change the CSS styles applied to a set of elements on the page.

• React to events, such as a mouse clicking a button.

• Validate the contents of a form before they are sent to the web server.

• Obtain information about the browser displaying the web page, such as the manufacturer and version, and even environmental information such as the current window size and local time.

• Display an alert in the user's browser

## 2.4.2 JavaScript Syntax

JavaScript has a simple syntax for writing statements, declaring variables, and adding comments. Any code you write must adhere to this syntax

## 2.4.3 Variables, Data types, Operators

Variables are used to store data. There are three ways to declare a variable:

1. Give it a name and a value.
2. Declare it without giving it a value by using the var keyword. Until a variable is given a value, JavaScript will return its value as undefined.
3. Combine the above two approaches (this is the recommended style).

There are two important rules for naming variables in JavaScript:

* 1. Variable names must begin with a letter or the underscore character.
  2. Variable names are case sensitive.

### 2.4.3.1 Data types

Unlike C#, Visual Basic, and other common programming languages, you cannot specify the type of a variable in JavaScript. You declare it with the var keyword, and then JavaScript attempts to discern its type for you. JavaScript recognizes three simple types:

* String: Any set of characters (alphanumeric and punctuation) enclosed in double quotes. To include special characters such as ",´, ;, \, and & in your string, escape them with a backslash. Also use a backslash to split a string over two or more lines.
* Number: Any integer or decimal number. Do not wrap a number between double quotes when you declare a numeric variable or it will be treated as a string.

JavaScript also converts data between types, which can lead to confusion if you are not careful. For example, the numeric value 0 evaluates to false in Boolean expressions, so it is important to use the correct operator when comparing the values of variables.

### 2.4.3 .3 Operators

An operator is a keyword or a symbol indicating how to combine one or more values into an expression.

For example, the addition operator + indicates that two numbers should be added together. There are six groups of operators in JavaScript:

4-Arithmetic operators indicate a mathematical function to be performed on values/variables

* + (addition)
* - (subtraction)
* \*(multiplication)
* / (division)
* %( modulus)
* ++ (increment)
* \_\_ (decrement)

5. Assignment operators assign values to JavaScript

6. Comparison operators determine if two values/variables are or are not equal. The first set of comparison operators converts the two values/variables to the same type before comparison.

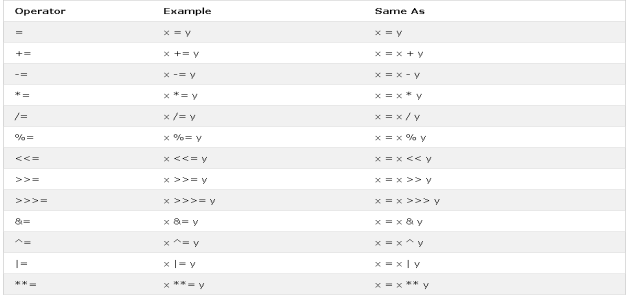


FIG 2.14 Operators

## 2.4.4 Functions

A function is a named sequence of statements that perform a specific task. Functions are useful for defining reusable blocks of code. After it has been defined, a function can be called from elsewhere in the script and the sequence of statements that constitute the function will run before execution returns to the next statement after the one that called the function.

Function definitions in JavaScript all have the same syntax

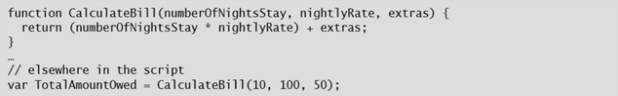


FIG 2.15 Creating and calling function

There are four parts to a function declaration:

• The function keyword indicates this is the start of a function definition.

• The function name. You use this name to run the function. It is case-sensitive.

• A comma-separated list of values, called arguments, which you can pass to the function.

This list is enclosed in parentheses. If the function has no arguments, it should still have the pair of parentheses after its name.

• A set of JavaScript statements enclosed in a pair of curly braces. These statements run when the function is invoked. A function uses the arguments like variables; it can read their values and modify them, but they only exist inside the function

# 2.5 OVERVIEW OF JQUER

## 2.5.1 What is the jQuery!?

jQuery is a lightweight, "write less, do more", JavaScript library.

The purpose of jQuery is to make it much easier to use JavaScript on your website.

jQuery takes a lot of common tasks that require many lines of JavaScript code to accomplish, and wraps them into methods that you can call with a single line of code.

jQuery also simplifies a lot of the complicated things from JavaScript, like AJAX calls and DOM manipulation.

The jQuery library contains the following features:

* HTML/DOM manipulation
* CSS manipulation
* HTML event methods
* Effects and animations
* AJAX
* Utilities

# 2.6 OVERVIEW OF BOOTSTRAP

## 2.6.1 What is the Bootstrap!?

Bootstrap is an open source product from Mark Otto and Jacob Thornton who, when

it was initially released, were both employees at Twitter. There was a need to standardize

the frontend toolsets of engineers across the company. In the launch blog post, Mark

Otto introduced the project like this:

In the earlier days of Twitter, engineers used almost any library they were familiar with

to meet front-end requirements. Inconsistencies among the individual applications made

it difficult to scale and maintain them. Bootstrap began as an answer to these challenges

and quickly accelerated during Twitter’s first Hackweek. By the end of Hackweek, we had

reached a stable version that engineers could use across the company.

Since Bootstrap launched in August 2011, it has taken off in popularity. It has evolved

from being an entirely CSS-driven project to include a host of JavaScript plugins and

icons that go hand in hand with forms and buttons. At its base, it allows for responsive

web design and features a robust 12-column, 940px-wide grid. One of the highlights is

the build tool on Bootstrap’s website, where you can customize the build to suit your

needs, choosing which CSS and JavaScript features you want to include on your site. All

of this allows frontend web development to be catapulted forward, building on a stable

foundation of forward-looking design and development. Getting started with Bootstrap

is as simple as dropping some CSS and JavaScript into the root of your site.

For someone starting a new project, Bootstrap comes with a handful of useful elements.

Normally, when I start a project, I start with tools like Eric Meyer’s Reset CSS and get

going on my web project. With Bootstrap, you just need to include the *bootstrap.css* CSS file and, optionally, the *bootstrap.js* JavaScript file into your website and you are ready to go.

**CHAPTER THREE**

(ANDROID)

# 3.1  WHAT'S ANDROID?

**Android is**

a mobile system developed by Google. It is used by several smart phones and tablets. Examples include the Sony X pair, the Samsung Galaxy, and the Google Nexus One. The Android operating system (OS) is based on the Linux kernel. Unlike Apple's IOS, Android is open source, meaning developers can modify and customize the OS for each phone. Therefore, different Android-based phones often have different graphical user interfaces GUIs even though they use the same OS.

# 3.2   WHAT'S OOP?

**Object-oriented programming (OOP)**

refers to a type of computer programming (software design) in which programmers define not only the data type of a data structure, but also the types of operations (functions) that can be applied to the data structure. In this way, the data structure becomes an object that includes both data and functions. In addition, programmers can create relationships between one object and another. For example, objects can inherit characteristics from other objects.

**3.3   WHAT'S GREAPHICAL USER INTERFACE?**

**The graphical user interface (GUI) is**

a form of user interface that allows users to interact with electronic devices through graphical icons and visual indicators such as secondary notation, instead of text-based user interfaces, typed command labels or text navigation.

# 3.4   WHAT'S JAVA TECNOLOGY AND WHY DO I NEED IT ?

**Java is**

a programming language and computing platform first released by Sun Microsystems in 1995. There are lots of applications and websites that will not work unless you have Java installed, and more are created every day. Java is fast, secure, and reliable. From laptops to datacenters, game consoles to scientific supercomputers, cell phones to the Internet, Java is everywhere!

# 3.5   IS JAVA FREE TO DOWNLOAD?

**Yes, Java is**

free to download. Get the latest version at java.com. If you are building an embedded or consumer device and would like to include Java, please Oracle for more information on including Java in your device. Why should I upgrade to the latest Java version? The latest Java version contains important enhancements to improve performance, stability and security of the Java applications that run on your machine. Installing this free update will ensure that your Java applications continue to run safely and efficiently.

# 3.6 WHAT WILL GET WHEN I DOWNLOAD JAVA SOFTWARE?

**The Java Runtime Environment (JRE) is**

what you get when you download Java software. The JRE consists of the Java Virtual Machine (JVM), Java platform core classes, and supporting Java platform libraries. The JRE is the runtime portion of Java software, which is all you need to run it in your Web browser.

# 3.7 WHAT'S JAVA PLUG-IN SOFTWARE?

**The Java Plug-in software is**

a component of the Java Runtime Environment (JRE). The JRE allows applets written in the Java programming language to run inside various browsers. The Java Plug-in software is not a standalone program and cannot be installed separately. I have heard the terms Java Virtual Machine and JVM. Is this Java software? The Java Virtual Machine is only one aspect of Java software that is involved in web interaction. The Java Virtual Machine is built right into your Java software download, and helps run Java applications

**3.8** **WHAT'S MODEL CLASS IN JAVA?**

**Programming patterns are**

good ideas (something bigger than code) that can be used over and over again in programming projects. The MVP pattern is a good way to think about writing all GUI applications: it separates the application into three main classes that interact with each other to coordinate the entire application

## How to implement MVP for Android

Well, this is where it all starts to become more diffuse. There are many variations of MVP and everyone can adjust the pattern to their needs and the way they feel more comfortable. It varies depending basically on the number of responsibilities that we delegate to the presenter.

Is the view responsible to enable or disable a progress bar, or should it be done by the presenter? And who decides which actions should be shown in the Action Bar? That’s where the tough decisions begin. I will show how I usually work, but I want this article to be more a place for discussion rather than strict guidelines on how to apply MVP, because up to there is no “standard” way to implement it.

For this article, I’ve implemented a very simple example that [you may find on my Github](https://github.com/antoniolg/androidmvp) with a login screen and a main screen. For simplicity purposes, the code in the article is in Kaitlin, but you can also check the code in Java 8 in [the repository](https://github.com/antoniolg/androidmvp).

**Model:**

in an application with a complete layered architecture, this model would only be the gateway to the domain layer or business logic. If we were using [Uncle Bob’s clean architecture](http://blog.8thlight.com/uncle-bob/2012/08/13/the-clean-architecture.html), the model would probably be an interact or that implements a use case. But for the purpose of this article, it is enough to see it as the provider of the data we want to display in the view.If you check the code, you will see that I’ve created two mock integrators with artificial delays to simulate requests to a server. The structure of one of this integrators: 

**Figure 3.1:Model in Android**

**view/controller classes:**

one might allow the user a much easier way to interact with the application.

**View:**

This class coordinates the appearance of the GUI. It decides where all the controls and displays go (labels, text fields, buttons, etc.) When notified by the model that the model has changed its state, the view redisplays itself by calling methods in the model that return all the information that the view must display. A view class is typically long and boring from a programming perspective, because it contains lots of specifics about how the GUI looks (things like fonts, sizes, foreground/background colors, and the placement of its components requires lots of details). Writing one involves understanding lots of classes from the standard Java library and inheritance

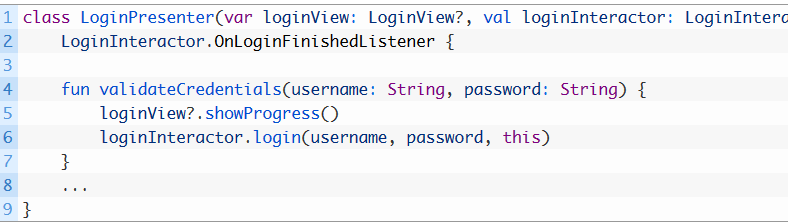


**Figure 3.2:View in Android**

**The presenter**

The presenter is responsible to act as the middleman between view and model. It retrieves data from the model and returns it formatted to the view

Also, unlike the typical MVC, it decides what happens when you interact with the view. So it will have a method for each possible action the user can do. We saw it in the view, but here’s the implementation:



**Figure 3.3:** presenter **in Android**

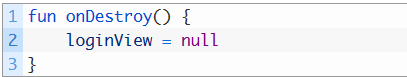
**MVP has some risks**, and the most important we use to forget is that the presenter is attached to the view forever. And the view is an activity, which means that:

* We can leak the activity with long-running tasks
* We can try to update activities that have already died

For the first point, if you can ensure that your background tasks finish in a reasonable amount of time, I wouldn’t worry much. Leaking an activity 5-10 seconds won’t make your App much worse, and the solutions to this are usually complex.

The second point is more worrying. Imagine you send a request to a server that takes 10 seconds, but the user closes the activity after 5 seconds. By the time the callback is called and the **UI is updated, it will crash because the activity is finishing**.

To solve this, we call the on Destroy() method that cleans the view:



**Figure 3.4: MVP in Android**

**3.9** **WHAT DOES JAVA OBJECT MEAN ?**

**A Java object is**

a combination of data and procedures working on the available data. An object has a state and behavior. The state of an object is stored in fields (variables), while methods (functions) display the object's behavior. Objects are created from templates known as classes. In Java, an object is created using the keyword "new" There are three steps to creating a Java object:

1. Declaration of the object 2. Instantiation of the object 3. Initialization of the object

When a Java object is declared, a name is associated with that object. The object is instantiated so that memory space can be allocated. Initialization is the process of assigning a proper initial value to this allocated space. The properties of Java objects include:

• One can only interact with the object through its methods. Hence, internal details are hidden. • When coding, an existing object may be reused. • When a program's operation is hindered by a particular object, that object can be easily removed and replaced.

A new object t from the class "tree" is created using the following syntax: Tree t = new Tree().

**3.10** **WHAT'S THE MAIN CLASS ?**

**In Java,**

you need to have a method named main in at least one class. The following is what must appear in a real Java program. ... In Java, main is a static method. This means the method is part of its class and not part of objects.

**3.11** **JAVA INHERITANCE (SUBCLASS AND SUPERCLASS)**

**In Java,**

it is possible to inherit attributes and methods from one class to another. We group the "inheritance concept" into two categories:

• subclass (child) - the class that inherits from another class • superclass (parent) - the class being inherited from

To inherit from a class, use the extends keyword.

In the example below, the Car class (subclass) inherits the attributes and methods from the Vehicle class (super class)

**CHAPTER FOUR**

(SERVER SIDE)

# 4.1 PHP ─ Introduction

PHP started out as a small open source project that evolved as more and more people

found out how useful it was. Rasmus Lerdorf unleashed the first version of PHP way backin 1994.

* PHP is a recursive acronym for "PHP: Hypertext Preprocessor".
* PHP is a server-side scripting language that is embedded in HTML. It is used to

manage dynamic content, databases, session tracking, even build entire ecommerce

sites.

* It is integrated with a number of popular databases, including MySQL, PostgreSQL,

Oracle, Sybase, Informix, and Microsoft SQL Server.

* PHP is pleasingly zippy in its execution, especially when compiled as an Apache

module on the Unix side. The MySQL server, once started, executes even very

complex queries with huge result sets in record-setting time.

* PHP supports a large number of major protocols such as POP3, IMAP, and LDAP.

PHP4 added support for Java and distributed object architectures (COM and

CORBA), making n-tier development a possibility for the first time.

* PHP is forgiving: PHP language tries to be as forgiving as possible.
* PHP Syntax is C-Like.

## 44.1.1Common Uses of PHP

PHP performs system functions, i.e. from files on a system it can create, open, read, write,

and close them. The other uses of PHP are:

* PHP can handle forms, i.e. gather data from files, save data to a file, thru emailyou can send data, return data to the user
* You add, delete, modify elements within your database thru PHP.
* Access cookies variables and set cookies.
* Using PHP, you can restrict users to access some pages of your website.
* It can encrypt data.

## 4.1.2Characteristics of PHP

Five important characteristics make PHP's practical nature possible:

* + Simplicity
  + Efficiency
  + Security
  + Flexibility
  + Familiarity

## 4.1.3"Hello World" Script in PHP

To get a feel of PHP, first start with simple PHP scripts. Since "Hello, World!" is an essential

example, first we will create a friendly little "Hello, World!" script.

As mentioned earlier, PHP is embedded in HTML. That means that in amongst your normal

HTML (or XHTML if you're cutting-edge) you'll have PHP statements like this:  
It will produce the following result:

<html>

<head>

<title>Hello World</title>

<body>

<?php echo "Hello, World!";?>

</body>

</html>

Hello, World!

# 4.2 PHP ─ Syntax Overview

## 4.2.1 Escaping to PHP

The PHP parsing engine needs a way to differentiate PHP code from other elements in the

page. The mechanism for doing so is known as 'escaping to PHP.' There are four ways to

do this:

Canonical PHP tags

The most universally effective PHP tag style is:

<?php...?>

If you use this style, you can be positive that your tags will always be correctly interpreted.

Short-open (SGML-style) tags

Short or short-open tags look like

this:

<?...?>

Short tags are, as one might expect, the shortest **option You must** do one of two things

to enable PHP to recognize the tags:

* Choose the --enable-short-tags configuration option when you're building PHP.
* Set the short\_open\_tag setting in your php.ini file to on. This option must be

disabled to parse XML with PHP because the same syntax is used for XML tags.

**ASP-style tags**

ASP-style tags mimic the tags used by Active Server Pages to delineate code blocks. ASPstyle

tags look like this:

<?...?>

To use ASP-style tags, you will need to set the configuration option in your php.ini file.

**HTML script tags**

HTML script tags look like this:

<script language="PHP">…...</script>

## 4.2.2 Commenting PHP Code

A *comment* is the portion of a program that exists only for the human reader and stripped

out before displaying the programs result. There are two commenting formats in PHP:

**Single-line comments:** They are generally used for short explanations or notes relevant

to the local code. Here are the examples of single line comments.

<?

# This is a comment, and

# This is the second line of the comment

// This is a comment too. Each style comments only

print "An example with single line comments";

?>

**Multi-lines printing:** Here are the examples to print multiple lines in a single print statement:

<?

# First Example

print <<<END

This uses the "here document" syntax to output

multiple lines with $variable interpolation. Note

that the here document terminator must appear on a

line with just a semicolon no extra whitespace!

END;

# Second Example

print "This spans

multiple lines. The newlines will be

output as well";

?>

**Multi-lines comments:** They are generally used to provide pseudocode algorithms and

more detailed explanations when necessary. The multiline style of commenting is the same

as in C. Here are the example of multi lines comments.

<?

/\* This is a comment with multiline

Author : Mohammad Mohtashim

Purpose: Multiline Comments Demo

Subject: PHP

\*/

print "An example with multi line comments";

?>

## 4.2.3 PHP is whitespace insensitive

Whitespace is the stuff you type that is typically invisible on the screen, including spaces,

tabs, and carriage returns (end-of-line characters).

PHP whitespace insensitive means that it almost never matters how many whitespace

characters you have in a row.one whitespace character is the same as many such

characters.

# 4.3 PHP ─ Variable Types

The main way to store information in the middle of a PHP program is by using a variable.

Here are the most important things to know about variables in PHP.

* All variables in PHP are denoted with a leading dollar sign ($).
* The value of a variable is the value of its most recent assignment.
* Variables are assigned with the = operator, with the variable on the left-hand side and the expression to be evaluated on the right.
* Variables can, but do not need, to be declared before assignment.
* Variables in PHP do not have intrinsic types - a variable does not know in advance
* whether it will be used to store a number or a string of characters.
* Variables used before they are assigned have default values.
* PHP does a good job of automatically converting types from one to another when

necessary.

* PHP variables are Perl-like.

**PHP has a total of eight data types which we use to construct our variables:**

**Integers:** are whole numbers, without a decimal point, like 4195.

**Doubles:** are floating-point numbers, like 3.14159 or 49.1.

**Booleans:** have only two possible values either true or false.

**NULL:** is a special type that only has one value: NULL.

**Strings:** are sequences of characters, like 'PHP supports string operations.'

**Arrays:** are named and indexed collections of other values.

**Objects:** are instances of programmer-defined classes, which can package up both

other kinds of values and functions that are specific to the class.

**Resources:** are special variables that hold references to resources external to PHP

(such as database connections).

The first five are *simple types*, and the next two (arrays and objects) are compound - the

compound types can package up other arbitrary values of arbitrary type, whereas the

simple types cannot.

We will explain only simile data type in this chapters. Array and Objects will be explained

separately.

## 4.3.1 Integers

They are whole numbers, without a decimal point, like 4195. They are the simplest

type. they correspond to simple whole numbers, both positive and negative. Integers can

be assigned to variables, or they can be used in expressions, like so:

$int\_var = 12345;

$another\_int = -12345 + 12345;

Integer can be in decimal (base 10), octal (base 8), and hexadecimal (base 16) format.

Decimal format is the default, octal integers are specified with a leading 0, and

hexadecimals have a leading 0x.

For most common platforms, the largest integer is (2\*\*31 . 1) (or 2,147,483,647), and

the smallest (most negative) integer is . (2\*\*31 . 1) (or .2,147,483,647).

## 4.3.2Doubles

They like 3.14159 or 49.1. By default, doubles print with the minimum number of decimal

places needed. For example, the code:

$many = 2.2888800;

$many\_2 = 2.2111200;

$few = $many + $many\_2;

print(.$many + $many\_2 = $few<br>.);

It produces the following browser output:

2.28888 + 2.21112 = 4.5

## 4.3.3 Boolean

They have only two possible values either true or false. PHP provides a couple of constants

especially for use as Booleans: TRUE and FALSE, which can be used like so:

if (TRUE)

print("This will always print<br>");

else

print("This will never print<br>");

**Interpreting other types as Booleans**

Here are the rules for determine the "truth" of any value not already of the Boolean type:

* + If the value is a number, it is false if exactly equal to zero and true otherwise.
  + If the value is a string, it is false if the string is empty (has zero characters) or is

the string "0", and is true otherwise.

* + Values of type NULL are always false.

If the value is an array, it is false if it contains no other values, and it is true

otherwise. For an object, containing a value means having a member variable that

has been assigned a value.

* + Valid resources are true (although some functions that return resources when they

are successful will return FALSE when unsuccessful).

* + Don't use double as Booleans.

Each of the following variables has the truth value embedded in its name when it is used

in a Boolean context.

$true\_num = 3 + 0.14159;

$true\_str = "Tried and true"

$true\_array[49] = "An array element";

CHAPTER **(4)**  Backend

## 4.3.4 NULL

NULL is a special type that only has one value: NULL. To give a variable the NULL value,

simply assign it like this:

$my\_var = NULL;

The special constant NULL is capitalized by convention, but actually it is case insensitive;

you could just as well have typed:

$my\_var = NULL;

A variable that has been assigned NULL has the following properties:

It evaluates to FALSE in a Boolean context.

It returns FALSE when tested with IsSet() function

## 4.3.5 Strings

They are sequences of characters, like "PHP supports string operations". Following are

valid examples of string:

$string\_1 = "This is a string in double quotes";

$string\_2 = "This is a somewhat longer, singly quoted string";

$string\_39 = "This string has thirty-nine characters";

$string\_0 = ""; // a string with zero characters

There are no artificial limits on string length - within the bounds of available memory, you

ought to be able to make arbitrarily long strings.

Strings that are delimited by double quotes (as in "this") are preprocessed in both the

following two ways by PHP:

Certain character sequences beginning with backslash (\) are replaced with special characters

Variable names (starting with $) are replaced with string representations of their values.

The escape-sequence replacements are:

\n is replaced by the newline character

\r is replaced by the carriage-return character

\t is replaced by the tab character

\$ is replaced by the dollar sign itself ($)

\" is replaced by a single double-quote (")

\\ is replaced by a single backslash (\)

## 4.3.6 Variable Naming

Rules for naming a variable is:

Variable names must begin with a letter or underscore character.

A variable name can consist of numbers, letters, underscores but you cannot use

characters like + , - , % , ( , ) . & ,etc

There is no size limit for variables.

## 4.3.7 PHP – Variables

Scope can be defined as the range of availability a variable has to the program in which it

is declared. PHP variables can be one of four scope types:

Local variables

Function parameters

Global variables

Static variables

# 4.4 PHP ─ Operator Types

**What is Operator?** Simple answer can be given using expression *4 + 5 is equal to 9*.

Here 4 and 5 are called operands and + is called operator. PHP language supports following

type of operators.

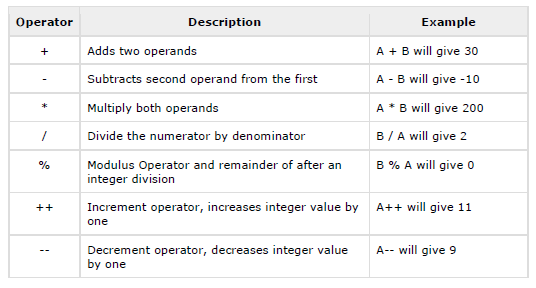
* + 1. Arithmetic Operators
    2. Comparison Operators
    3. Logical (or Relational) Operators
    4. Assignment Operators
    5. Conditional (or ternary) Operators

Let’s have a look on all operators one by one.

## 4.4.1 Arithmetic Operators

The following arithmetic operators are supported by PHP language:

Assume variable A holds 10 and variable B holds 20 then:



**Example**

Try the following example to understand all the arithmetic operators. Copy and paste

following PHP program in test.php file and keep it in your PHP Server's document root and

browse it using any browser.

<html>

<head><title>Arithmetical Operators</title><head>

<body>

<?php

$a = 42;

$b = 20;

$c = $a + $b;

echo "Addition Operation Result: $c <br/>";

$c = $a - $b;

echo "Subtraction Operation Result: $c <br/>";

$c = $a \* $b;

echo "Multiplication Operation Result: $c <br/>";

$c = $a / $b;

echo "Division Operation Result: $c <br/>";

$c = $a % $b;

echo "Modulus Operation Result: $c <br/>";

$c = $a++;

echo "Increment Operation Result: $c <br/>";

$c = $a--;

This will produce the following result:

Addition Operation Result: 62

Subtraction Operation Result: 22

Multiplication Operation Result: 840

Division Operation Result: 2.1

Modulus Operation Result: 2

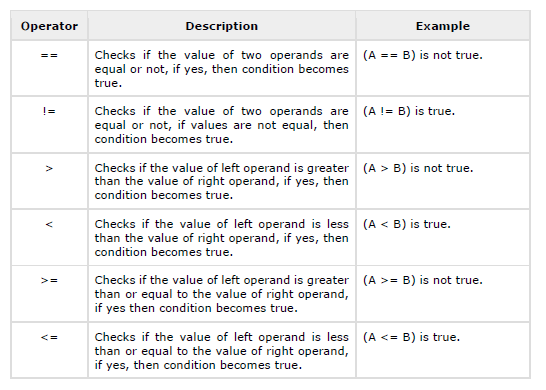
Increment Operation Result: 42

Decrement Operation Result: 43

## 4.4.2 Comparison Operators

There are following comparison operators supported by PHP language.

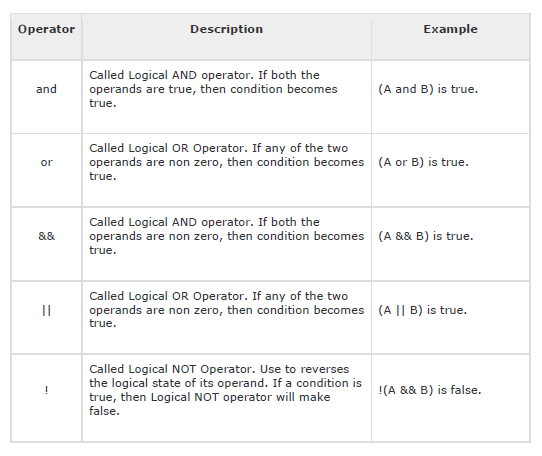
Assume variable A holds 10 and variable B holds 20 then:



## 4.4.3 Logical Operators

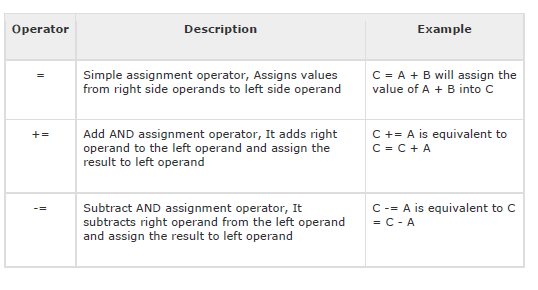
The following logical operators are supported by PHP language.

Assume variable A holds 10 and variable B holds 20 then:



## 4.3.4 Assignment Operators

PHP supports the following assignment operators:



## 4.3.5 conditional Operators



# 4.4 PHP ─ Decision Making

The if, else if ...else and switch statements are used to take decision based on the different

condition.

You can use conditional statements in your code to make your decisions. PHP supports the

following three decision making statements:

**if...else statement** - use this statement if you want to execute a set of code when

a condition is true and another if the condition is not true.

**else if statement** - is used with the if...else statement to execute a set of code if

**one** of several condition are true.

**switch statement** - is used if you want to select one of many blocks of code to be

executed, use the Switch statement. The switch statement is used to avoid long

blocks of if..else if. code.

## 4.4.1 The If...Else Statement

If you want to execute some code if a condition is true and another code if a condition is

false, use the if....else statement.

**Syntax**

if (condition)

code to be executed if condition is true;

else

code to be executed if condition is false;

## 4.4.2 The Else If Statement

If you want to execute some code if one of the several conditions is true, then use the

else if statement.

**Syntax**

if (condition)

code to be executed if condition is true;

elseif (condition)

code to be executed if condition is true;

else

code to be executed if condition is false;

## 4.4.3 The Switch Statement

If you want to select one of many blocks of code to be executed, use the Switch statement.

The switch statement is used to avoid long blocks of if..elseif..else code.

**Syntax**

switch (expression)

{

case label1:

code to be executed if expression = label1;

break;

# 4.5 PHP ─ Loop Types

Loops in PHP are used to execute the same block of code a specified number of times. PHP

supports following four loop types.

* + **for -** loops through a block of code a specified number of times.
  + **while -** loops through a block of code if and as long as a specified condition is true.
  + **do...while -** loops through a block of code once, and then repeats the loop as long

as a special condition is true.

* + **for each -** loops through a block of code for each element in an array.

## 4.5.1 The for loop statement

The for statement is used when you know how many times you want to execute a

statement or a block of statements

**Syntax**

for (initialization; condition; increment)

{

code to be executed;

}

The initialize is used to set the start value for the counter of the number of loop iterations.

A variable may be declared here for this purpose and it is traditional to name it $i.

## 4.5.2 The while loop statement

The while statement will execute a block of code if and as long as a test expression is true.

If the test expression is true, then the code block will be executed. After the code has

executed the test expression will again be evaluated and the loop will continue until the

test expression is found to be false.

**Syntax**

while (condition)

{

code to be executed;

}

## 4.5.3 The do...while loop statement

The do...while statement will execute a block of code at least once - it will then repeat the

loop as long as a condition is true.

**Syntax**

do

{

code to be executed;

}while (condition);

## 4.5.4 The for each loop statement

The foreach statement is used to loop through arrays. For each pass the value of the

current array element is assigned to $value and the array pointer is moved by one and in

the next pass next element will be processed.

**Syntax**

foreach (array as value)

{

code to be executed;

}

## 4.5.5 The break statement

The PHP **break** keyword is used to terminate the execution of a loop prematurely.

The **break** statement is situated inside the statement block. If gives you full control and

whenever you want to exit from the loop you can come out. After coming out of a loop

immediate statement to the loop will be executed.

**Example**

In the following example, the condition test becomes true when the counter value reaches

3 and loop terminates.

<html>

<body>

<?php

$i = 0;

while( $i< 10)

{

$i++;

if( $i == 3 )break;

}

echo ("Loop stopped at i = $i" );

?>

</body>

This will produce the following result:

Loop stopped at i = 3



## 4.5.6 The continue statement

The PHP **continue** keyword is used to halt the current iteration of a loop but it does not

terminate the loop.

Just like the **break** statement the **continue** statement is situated inside the statement

block containing the code that the loop executes, preceded by a conditional test. For the

pass encountering **continue** statement, rest of the loop code is skipped and next pass starts.

**Example**

In the following example, the loop prints the value of array, but when the condition

becomes true, it just skips the code and next value is printed.

<html>

<body>

<?php

$array = array( 1, 2, 3, 4, 5);

foreach( $array as $value )

{

if( $value == 3 )continue;

echo "Value is $value <br />";

}

?>

</body>

</html>

This will produce the following result:

Value is 1

Value is 2

Value is 4

Value is 5



**CHAPTER FIVE**

(DATA BASE)

**Conclusion:**

We gained a lot of experiences as a team working on this project, we did our best to make a useful application that the people can use, working with a team in this graduation project has been a pleasure for all us specially under the super

vision of our supervisor “DR. Eid Abd-Elhakim”

**Reference :**

• https://developer.android.com/

• https://www.android.com/

• https://en.wikipedia.org/

• https://github.com/

• https://android-arsenal.com/

• https://www.datacamp.com/

• https://www.guru99.com

• https://php.net/

• https://css-tricks.com/

• https://jquery.com/

• https://jupyter.org/

• https://www.oracle.com

**.**https://trello.com/b/Mb9numcu/graduation-project