

CLASS XI



WEB APPLICATIONS

LEVEL - 3

Student Handbook



Central Board of Secondary Education
Shiksha Kendra, 2, Community Centre, Preet Vihar, Delhi-110301





Web Applications

Student Handbook

Class XI



CENTRAL BOARD OF SECONDARY EDUCATION
Shiksha Kendra, 2, Community Centre, Preet Vihar, Delhi-110301

Web Applications
Student Handbook Class XI

PRICE : ₹

FIRST EDITION

COPIES :

No Part of this publication may be reproduced, stored in a retrieval system or transmitted, in any form or by any means, electronic, mechanical photocopying, recording or otherwise without the prior permission of the publisher.

PUBLISHED BY : The Secretary, Central Board of Secondary Education,
Shiksha Kendra, 2, Community Centre, Preet Vihar,
Delhi - 110301

DESIGN & LAYOUT BY : India Offset Press, A-1, Mayapuri Industrial Area, Ph-1, ND-64

PRINTED BY : M/s.



भारत का संविधान

उद्देशिका

हम, भारत के लोग, भारत को एक सम्पूर्ण ¹प्रभुत्व-संपन्न समाजवादी पंथनिरपेक्ष लोकतंत्रात्मक गणराज्य बनाने के लिए, तथा उसके समस्त नागरिकों को:

सामाजिक, आर्थिक और राजनैतिक न्याय,

विचार, अभिव्यक्ति, विश्वास, धर्म

और उपासना की स्वतंत्रता,

प्रतिष्ठा और अवसर की समता

प्राप्त कराने के लिए

तथा उन सब में व्यक्ति की गरिमा

²और राष्ट्र की एकता और अखंडता

सुनिश्चित करने वाली बंधुता बढ़ाने के लिए

दृढ़संकल्प होकर अपनी इस संविधान सभा में आज तारीख 26 नवम्बर, 1949 ई० को एतद्वारा इस संविधान को अंगीकृत, अधिनियमित और आत्मार्पित करते हैं।

-
1. संविधान (बयालीसवां संशोधन) अधिनियम, 1976 की धारा 2 द्वारा (3.1.1977) से “प्रभुत्व-संपन्न लोकतंत्रात्मक गणराज्य” के स्थान पर प्रतिस्थापित।
 2. संविधान (बयालीसवां संशोधन) अधिनियम, 1976 की धारा 2 द्वारा (3.1.1977) से “राष्ट्र की एकता” के स्थान पर प्रतिस्थापित।
-

भाग 4 क

मूल कर्तव्य

51 क. मूल कर्तव्य – भारत के प्रत्येक नागरिक का यह कर्तव्य होगा कि वह –

- (क) संविधान का पालन करे और उसके आदर्शों, संस्थाओं, राष्ट्रध्वज और राष्ट्रगान का आदर करे;
- (ख) स्वतंत्रता के लिए हमारे राष्ट्रीय आंदोलन को प्रेरित करने वाले उच्च आदर्शों को हृदय में संजोए रखे और उनका पालन करे;
- (ग) भारत की प्रभुता, एकता और अखंडता की रक्षा करे और उसे अक्षुण्ण रखे;
- (घ) देश की रक्षा करे और आह्वान किए जाने पर राष्ट्र की सेवा करे;
- (ङ) भारत के सभी लोगों में समरसता और समान भ्रातृत्व की भावना का निर्माण करे जो धर्म, भाषा और प्रदेश या वर्ग पर आधारित सभी भेदभाव से परे हों, ऐसी प्रथाओं का त्याग करे जो स्त्रियों के सम्मान के विरुद्ध हैं;
- (च) हमारी सामासिक संस्कृति की गौरवशाली परंपरा का महत्त्व समझे और उसका परिरक्षण करे;
- (छ) प्राकृतिक पर्यावरण की जिसके अंतर्गत वन, झील, नदी, और वन्य जीव हैं, रक्षा करे और उसका संवर्धन करे तथा प्राणी मात्र के प्रति दयाभाव रखे;
- (ज) वैज्ञानिक दृष्टिकोण, मानववाद और ज्ञानार्जन तथा सुधार की भावना का विकास करे;
- (झ) सार्वजनिक संपत्ति को सुरक्षित रखे और हिंसा से दूर रहे;
- (ञ) व्यक्तिगत और सामूहिक गतिविधियों के सभी क्षेत्रों में उत्कर्ष की ओर बढ़ने का सतत प्रयास करे जिससे राष्ट्र निरंतर बढ़ते हुए प्रयत्न और उपलब्धि की नई उंचाइयों को छू ले;
- ¹(ट) यदि माता-पिता या संरक्षक हैं, छह वर्ष से चौदह वर्ष तक की आयु वाले अपने, यथास्थिति, बालक या प्रतिपाल्य के लिये शिक्षा के अवसर प्रदान करे।

-
1. संविधान (छयासीवां संशोधन) अधिनियम, 2002 की धारा 4 द्वारा प्रतिस्थापित।

THE CONSTITUTION OF INDIA

PREAMBLE

WE, THE PEOPLE OF INDIA, having solemnly resolved to constitute India into a ¹**SOVEREIGN SOCIALIST SECULAR DEMOCRATIC REPUBLIC** and to secure to all its citizens :

JUSTICE, social, economic and political;

LIBERTY of thought, expression, belief, faith and worship;

EQUALITY of status and of opportunity; and to promote among them all

FRATERNITY assuring the dignity of the individual and the² unity and integrity of the Nation;

IN OUR CONSTITUENT ASSEMBLY this twenty-sixth day of November, 1949, do **HEREBY ADOPT, ENACT AND GIVE TO OURSELVES THIS CONSTITUTION.**

1. Subs, by the Constitution (Forty-Second Amendment) Act. 1976, sec. 2, for "Sovereign Democratic Republic" (w.e.f. 3.1.1977)

2. Subs, by the Constitution (Forty-Second Amendment) Act. 1976, sec. 2, for "unity of the Nation" (w.e.f. 3.1.1977)

THE CONSTITUTION OF INDIA

Chapter IV A

FUNDAMENTAL DUTIES

ARTICLE 51A

Fundamental Duties - It shall be the duty of every citizen of India-

- (a) to abide by the Constitution and respect its ideals and institutions, the National Flag and the National Anthem;
- (b) to cherish and follow the noble ideals which inspired our national struggle for freedom;
- (c) to uphold and protect the sovereignty, unity and integrity of India;
- (d) to defend the country and render national service when called upon to do so;
- (e) to promote harmony and the spirit of common brotherhood amongst all the people of India transcending religious, linguistic and regional or sectional diversities; to renounce practices derogatory to the dignity of women;
- (f) to value and preserve the rich heritage of our composite culture;
- (g) to protect and improve the natural environment including forests, lakes, rivers, wild life and to have compassion for living creatures;
- (h) to develop the scientific temper, humanism and the spirit of inquiry and reform;
- (i) to safeguard public property and to abjure violence;
- (j) to strive towards excellence in all spheres of individual and collective activity so that the nation constantly rises to higher levels of endeavour and achievement;
- ¹(k) to provide opportunities for education to his/her child or, as the case may be, ward between age of 6 and 14 years.

1. Subs. by the Constitution (Eighty - Sixth Amendment) Act, 2002

Preface

*I*n an increasingly globalised world and the changing paradigm of urbanized and rural living the demand for Information Technology (IT) has been increased manifold throughout the world. In this ever expanding sector, it has become essential to provide competency based Vocational Education. It is in this context that CBSE has launched a course in Information Technology under NVEQF (now NSQF) from level 1 to 4.

This Student Handbook, Web Applications for class XI, which forms a part of vocational Qualification package for students, was prepared by expertise in the field. The IT-ITeS Skill Development Council approved by the National Skill Development Corporation (NSDC) for the IT/ITeS Industry developed the National Occupation Standards (NOS). The National Occupation Standards are a set of competency standards and guidelines endorsed by the representatives of IT Industry for recognizing and assessing skills and knowledge needed to perform effectively in the workplace.

It has been a deliberate effort to keep the language used in this student handbook as simple as possible for the benefit of the students. Necessary pictorial illustrations and tables have been included to help the students to understand the concepts without any difficulty.

Practicing professionals from the field of Information Technology (IT) comprised the team of authors for this book. I hope this book will help the students to serve as useful resource.

The Board is grateful to the members of the Committee of Course for their advice, guidance and total commitment towards development of this course. We are indeed indebted to these academic advisors who have lent us the benefit of their rich and insightful experience. I would like to appreciate Vocational Education Cell, CBSE for coordinating and successfully completing the work.

Comments and suggestions are welcome for further improvement of the book.

Chairman, CBSE



Acknowledgements

— Advisors —

- **Shri Y.S.K. Seshu Kumar**, Chairman, CBSE
- **Shri K.K. Choudhury**, Controller of Examination & Director (Vocational Education), CBSE

— Authors —

- **Dr. Manoj Kumar**, Associate Professor & HOD (CSE) Convener & Author
Ambedkar Institute of Advanced Communication Technologies & Research,
Govt. of NCT of Delhi
- **Ms. Manju Khari**, Assistant Professor (CSE)
Ambedkar Institute of Advanced Communication Technologies & Research,
Govt. of NCT of Delhi
- **Dr. Pradeep Kumar**, Associate Professor
Maulana Azad National Urdu University (MANUU), Hyderabad

— Co-ordination —

- **Dr. Biswajit Saha**, Additional Director, Vocational Education, CBSE



Contents

1. Multimedia Authoring – Animation Tools	1
Section 1: Animation Concept	1
Section 2: Frames and Layers.....	9
Section 3: Motion and Shape Tweening	13
Section 4: Importing AV File	20
Section 5: Multimedia Publishing.....	23
2. Digital Content Creation – Adding Styles to Web Pages(CSS)	27
Section 1: Basics About HTML	27
Section 2: Basics About CSS	38
Section 3: CSS Properties – border, box font, margin, padding CSS classes ..	45
Section 4: Publishing Your Website or Webpages	57
3. Web Scripting JavaScript	61
Section 1: Creating interactive Web Pages with Scripts	61
Section 2: Variables and Operators	76
Section 3: Decision making using if and switch	85
Section 4: Iterations: Loops	94
Section 5: Window Object	105
Section 6: Popup Boxes	118
Section 7: Arrays	122
Section 8: Functions	131

4. Work Integrated Learning IT - WA-I	140
5. Case Study: Designing a web-site for your school	143



1

MULTIMEDIA AUTHORIZING – ANIMATION TOOLS

Learning Objectives

After studying this unit the students will be able to:

Section-1: What is animation

Section-2: What is Frame & Layers

Section-3: Motion and Shape tweening

Section-4: Importing AV files

Section-5: Publishing work in animation

Section-1: Animation Concepts

1.1 Introduction

Computer Animation

It will not be an exaggeration to say that animation can bring the dullest of the features to life. It has the magic of injecting energy and emotions into the most seemingly inanimate objects. Over the years the advancement of technology has made animation, a very attractive and much sought after the component of multimedia.

What is Animation?

‘To animate’ literally means to give life to. Animating is moving something that cannot move on its own. Animation adds the dimensions of time to graphics which tremendously increase the potential of transmitting the desired information. In order to animate

something the animator has to be able to specify directly or indirectly how the 'thing' has to move through time and space.

The Heritage of Animation

It's interesting to note that the concept of animation has been a focus of study and research for a long time. Resources indicate that as far back as in 1824, Peter Roget presented a paper titled "The persistence of vision with regard to moving objects" to the British Royal Society. Later in 1887, famous scientist Thomas Edison started his research into motion pictures. Two years later, he announced his kinetoscope, which projected a 50ft length of film in approximately 13 seconds. Winsor McCay is considered by many to produce the first popular animation. He used rice paper to draw images each were slightly different from the other. Later, John Bray in 1910 used translucent cels (short for celluloid) in compositing multiple layers of drawings into a final image as well as the use of grey scale (as opposed to black and white) drawings. Later, In 1915, Fleischer patented 'roto scoping.' Roto scoping is drawing images on cells by tracing over previously recorded live action. Many such innovative ideas were put to test before Walt Disney burst into the scene with his film "Alice in Wonderland" in which live action was combined with cartoon characters. Later, In 1928, Walt Disney created the first cartoon with synchronized sound called "Mickey Mouse." Some of the innovative techniques used by Disney were the use of storyboard to review the story, the use of pencil sketches to review motion and the multi-plane camera stand. Multi-plane camera stand is a powerful technique, proposed in 1998 in Patent which allows a sort of parallax effect, moving the backgrounds at different rates as the observer pans across an environment to create an illusion of depth and zooming. In 1964, the scene of animation shifted to computers for the first time when Ken Knowlton, while working at Bell Laboratories, started developing computing techniques for producing animated movies. The University of Utah is said to be amongst the earlier pioneers in computer graphics. In the late sixties, Chuck Csuri of the Ohio State University did some initiatory work in computer animation as a member of the Computer Graphics Research Group. In the early 80s, the research group became the Advanced Computing Center for Art and Design and continues to produce computer animation. Since then, the scene of computer animation has remained dynamic with new ideas and path-breaking innovations. Some of the current activity centers include University of Toronto's computer Science Department, UC San Diego's Department of Computer Science and Engineering University of North Carolina's Computer Science Department and MIT's Media Lab and MIT's Lab for Computer Science.

1.2 Types of Animation

- ◆ **Procedural:** Animation is generated by writing a program that spits out the position/shape/whatever of the scene over time
- ◆ **Representation:** This technique allows an object to change its shape during the animation. There are three subcategories: The first is the animation of articulated objects, i.e., complex objects composed of connected rigid segments. The second is soft object animation used for deforming and animating the deformation of objects, e.g. skin over a body or facial muscles. The third is morphing which is the changing of one shape into another quite different shape. This can be done in two or three dimensions.
- ◆ **Stochastic Animation:** This uses stochastic processes to control groups of objects, such as in particle systems. Examples are fireworks, fire, water falls, etc.
- ◆ **Behavioral Animation:** Objects or “actors” are given rules about how they react to their environment. Examples are schools of fish or flocks of birds where each individual behaves according to a set of rules defined by the animator.
- ◆ **2D animation:** Two dimensional animation
- ◆ **3D animation:** Uses three dimensional object for creating animation

1.3 Traditional Methods

As evident from the history, animators used and invented a variety of different animation techniques. Traditionally, most of the animation was done by hand. All the frames in an animation had to be drawn by hand. Since each second of animation requires 24 frames (film), the amount of work required to create even the shortest of movies, can be tremendous. Some of the traditionally used methods are described below.

Key Frames

In this technique a storyboard is laid out and then the artists draw the major frames of the animation. These major frames are, those in which a lot of changes take place. They are the key points of animation. Later, a bunch of artists drawn the frames in between. This technique is, of course, very time and effort intensive.

Cel Animation

In this technique, each character is drawn on a separate piece of opaque paper. Then, at the time of shooting animation, the different characters are overlaid on top of the background in each frame. This is relatively a less tedious process, as the artists do not

have to draw the entire frames but just the parts that need to change such as individual character.

Rota scoping

Rota scoping is a technique where images are copied from a moving video into an animation. The animator draws the motion and shape of the object by referring to the video as opposed to imagining in his head. With the help of the rota scoping, one can animate some complex scenes that would be hard to visualize otherwise. The disadvantage is that one will have to hunt for the exact video that one wants to animate.

Computer Animation

With time the technique of animation has become more and more computer assisted and computer generated. All such techniques require a trade-off between the level of control that the animator has over the finer details of the motion and the amount of work that the computer does on its own. Broadly, the computer animation falls into three basic categories: key framing, motion capture, and simulation.

Key framing

The significance of the term “Key framing” can be traced back to traditional hand animation technique. Key framing requires the animator to specify critical or key positions for the objects. The computer then automatically fills in the missing frames by smoothly interpolating between those positions. The characters for a movie called Toy Story that was made in 1995, were designed through key frame animation. It's believed that each character had as many as 700 controls. Key framing requires that the animator has well thought of plan about how the moving objects are going to behave over time as well as the talent to express that information through key framed information. The continued popularity of key framing is a function of the degree of control that it allows the animator to exercise over the subtle details of the motion.

Motion Capture

Another technique is Motion Capture, in which magnetic or vision-based sensors record the actions of a human or animal object in three dimensions. A computer then uses these data to animate the object. This technology has enabled a number of famous athletes to supply the actions for characters in sports video games. Motion capture is pretty popular with the animators mainly because some of the commonplace human actions can be captured with relative ease. However, there can be serious discrepancies between the

shapes or dimensions of the subject and the graphical character and this may lead to problems of exact execution.

Simulation

Unlike key framing and motion picture, simulation uses the laws of physics to generate motion of figures and other objects. Virtual humans are usually represented as a collection of rigid body parts. These models though physically plausible are only an approximation of the human body. With more research and complex models the simulations are becoming increasingly life like. Simulations can be easily used to produce slightly different sequences in maintaining physical realism while in other animations like key framing or motion capture, a mere speeding up or slowing down the playback can spoil the naturalness of motion. Secondly, real-time simulations allow higher degree of interactivity where the real person can manoeuvre the actions of the simulated character. In contrast the applications based on key framing and motion select and modify motions form a pre computed library of motions. One drawback that simulation suffers from is the expertise and time required to handcraft the appropriate controls systems.

1.4 Hardware and Software

Hardware and software are the two major factors that determine the quality of computer animation that is produced. As both of these components work in conjunction with each other. It is important to make the right choice for a product that is of lasting quality and accuracy.

1.4.1 Hardware

Hardware comes in many shapes, sizes, and capabilities. Some hardware is specialized to do only certain tasks while other hardware are equipped for a variety of things. Some of the commonly used hardware are:

Silicon Graphics Inc. (SGI)

The SGI platform is one of the most widely used platforms for quality computer animation productions. SGI computers operate using the wide spread UNIX system. Produced by Silicon Graphics, these computers are extremely fast and produce excellent results. They come in a variety of types, ranging from the general purpose Indy to high power Indigo Extreme that is used for animations. Onyx is another type, which is suited to the complex calculations involved in rendering. Some of the software like Wave front, Alias, and Soft Image that run on SGI's.

PC's

PC's are very versatile machines combining flexibility and power. PC's have proven to be very useful for small companies and businesses as platforms to do computer animation. Applications such as 3DStudio and Animator Studio are used on PC's to make animations.

Macintosh

Macintosh or Mac were originally designed for graphic and desktop publishing and hence are pretty useful platforms for producing computer graphics and animation software. Some of these applications that work well on Mac's are Adobe Products like Photoshop and Premiere and Strata with Strata Studio Pro.

Amiga

Originally owned by Commodore, Amiga computers have held a position in the computer animation for long. The two software packages that Amiga is associated with are: Video Toaster and Light Wave 3D. The new d'Amiga systems has been customized to be a great graphics machine. Sophisticated hardware has to be coupled with a good software to produce good results. There are literally hundreds of computer animation and graphic software packages. However, only some are considered good enough.

1.4.2 Software

Some of the popular software packages used by the companies, schools and individuals all around the globe include some of the following.

3Dstudio Max

The successor to 3DStudio 3.0. 3DStudio Max runs under Windows NT. It is entirely object oriented, featuring new improvements, such as volumetric lighting, space warps, and all new redesigned interface.

LightWave3D

LightWave3D is another high end PC 3D computer software package. Originally developed for the Amiga platform, it is considered as the best 3D product for the PC. This software has been used for quite a few television productions such as Babylon 5 and Sea Quest.

Adobe Premiere

Adobe Premiere is a tool that is used to composite Digitized video, stills, and apply a variety of transitions and special effects. Adobe Premiere runs both on Macintoshes and PC Windows.

Alias I Wave front

Alias is one of the topmost animation packages in the market. Produced by two companies, Alias and Wave front who work in collaboration, it runs on the SGI's. Alias is well known for its great modeller which is capable of modelling some of the most complicated objects. Also, this software package is very flexible allowing programmers that will allow the programmers to create software that will run in tandem.

Animator Studio

Animator Studio is a cell animation program from Auto Desk. Animator runs on Windows. It has a multitude of features that minimizes the animation creation time.

Elastic Reality

This is one of the top line programs used for morphing. Elastic Reality runs on Mac and SGI's. One of the distinctive features of Elastic Reality is that it uses splines as opposed to points to define the morphing area. Elastic Reality allows to morph video as well as still images.

Soft Image

Soft Image is one of the three top most computer animation software packages. Soft Image is used in many top production studios around the country and around the world.

Strata Studio Pro

Strata Studio Pro is probably known as most popular 3D graphics application on the Mac. Created by Strata Inc. Strata Studio Pro is mainly a still graphic rendering application, but it does have animation capabilities also.

Director Vs. Flash

Two of the currently popular and widely used tools are; Director and Flash. Both are the products of the same company but interestingly enough have developed unique audiences often working in separate even opposing campuses. The following section of the paper traces the reasons that make both these products so competitive in the field of multimedia.

Flash

Flash on the other hand was built for the Web and was purchased by Macromedia in the mid-90s from a company called FutureSplash. The product earlier known as the Animator was rechristened as Flash. Flash is based on scalable vector graphics instead of fixed size bitmapped images, and a growing number of Web builders have adopted Flash as a somewhat replacement for HTML. The Shockwave Player, in contrast, doesn't lend itself well to full window presentations because of the bitmapped Director graphics. Flash graphics can bend and stretch and still look gorgeous, but bitmapped graphics end up distorted when manipulated. Director content is usually developed for a fixed size and the included as a supplement to standard HTML text.

Streaming differences

In the beginning streaming was a big advantage of Flash over Director. However the playing field has been levelled recently. Director now has streaming capabilities. From the user's point of view there is not a big difference between the streaming performance of Flash and Director Content. The quality of playback in both the tool is about the same.

Animation Styles and Asset Management

As mentioned earlier, Flash artwork can be used as symbols but Director automatically turns everything into a cast member. Because it functions as an illustration and an animation program, Flash can afford more flexibility. The lines and shapes of animation sequence in Flash are easily editable without too many obstructions. On the other hand, using bitmaps and having limited drawing tools like Director cannot support free-form animation. Thus, Flash provides a more interactive animation platform.

The Interfaces

Flash and Director have interfaces similar in function but quite different in appearance and implementation. Director has a very strictly defined authoring metaphor of Cast, Score and Stage. All objects including pictures, sounds and behaviours used in a movie are called Cast Members and are centrally organized in the window. Time based activity is laid out and edited using the Score window, which allows only one object per layer. On the other hand Flash has a slightly fuzzy authoring metaphor. Artwork may or may not be designated as symbols. The Timeline places no restrictions on the number of object combinations that can occupy a single layer, and scripting can be organized according to the developer's team.

Section 1: Exercise Questions

- Q. 1: What do you understand by the word animation?
- Q. 2: What are the key frames?
- Q. 3: What do you understand by SGI?
- Q. 4: What is amiga?
- Q. 5: Name any one of the topmost animation packages in the market.
- Q. 6: What are the various types of animation?
- Q. 7: Components like fireworks, fire, water falls, etc. can be implemented via which animation?
- Q. 8: Name the most well-known 3D graphics application on the Mac?
- Q. 9: Fill in the blanks
 - (a) is a technique where images are copied from a moving video into an animation.
 - (b) is a technique in which magnetic or vision-based sensors record the actions of a human or animal object in three dimensions.
 - (c) In the Animation is generated by writing a program that spits out the position/shape/whatever of the scene over time.

Further Readings

- 1. <http://en.wikipedia.org/wiki/Multimedia>
- 2. <http://svitmms.wikispaces.com/file/view/AuthoringSoftware.pdf>
- 3. <http://www.compuphase.com/animtips.htm>

Section-2: Frames and Layers

In this topic under Multimedia Authoring – Animation Tools, you will gain knowledge on theoretical and practical about the implementation of frames and layers in animation in flash.

2.1 Creating Frames and Key frames

The Timeline is the key to animation. As you look at the Timeline you'll notice the numbers across the top and the layers along the left side as shown in figure1. Each number has a matching frame. These frames are white when they are empty and turn gray when they contain content.

To create animation, you'll be manipulating two types of frames: Key frames and regular frames. A Key frame is a point in your movie where new content first appears. They are the key to animation. In **Key frames** you can also scale and reposition elements. The first frame of each layer is a Key frame. You can always identify the Key frames because they contain a tiny filled circle as shown in figure 1.1.

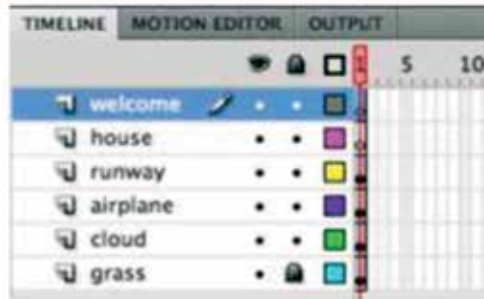


Fig. 1.1: Time Line (Tool “Flash”)

Regular **frames** are static and contain the same content as the preceding Key frame. They are sometimes called “slaves” to the Key frame. Frames contain a hollow circle clearly depicted in figure 1.2 and 1.3.

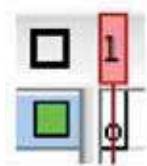


Fig.1.2: Frame

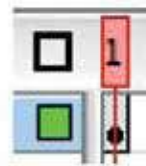


Fig. 1.3: Key Frame

Right now our movie only contains one frame, a Key frame. You'll want to estimate how many frames you think you'll need in your movie. We're going to create an animation that's about 10 seconds in length. Most developers use a **frame rate** of 12 frames per second (fps) which is the default in Flash. To develop a movie of 10 seconds and frame rate at 12 fps, we'll need 120 frames. We can always add or remove frames if we decide to change the length of our movie.

To insert frames on the Timeline

Click on frame 120 of the welcome layer. Look for the grey band at to find frame 120; click on it. Shift-click on each of the other layers down to the cloud layer. As you shift-click on each layer it will turn gray. Pull down the Insert menu, choose Timeline, and select Frame. The frames will all be shaded with grey because each layer contains content. The last frame (120) will contain a white rectangle.

2.2 Creating Key frames

Right now, all the elements appear on the first frame and nothing happens for the entire length of the movie (120 frames). Not every exciting, huh? We'll add Key frames at points in the movie where we want the content to change. For example, we want the runway to disappear after the plane has taken off and we don't want the grass or house to show until the plane has reached the end of the runway. We might want the cloud to appear, disappear, and reappear in a different position giving the feeling that the airplane is moving across the sky. At this point, we can only guess where we want these changes to take place. Key frames are easy to adjust if we decide we want to change the timing of different elements.

Let's start with the Welcome layer containing our text. Rather than showing the question "Have you ever wondered how airplanes fly?" at the beginning of the movie, let's just show it during the last 20 seconds of our movie. We'll make a new Key frame at Frame 100. Then, delete the content from Frame 1.

To add a Key frame

Click Frame 100 of the Welcome layer as shown in figure 1.4. Sometimes, it's difficult to tell which frame you're selecting. The status bar at the bottom of the Timeline shows the selected frame. Pull down the Insert menu, choose Timeline, select Key frame. You can also use the F6 key to insert a Key frame. A solid black circle will appear in Frame 100. Notice that there's now a line between Frames 99 and 100. A rectangle appears in Frame 23 indicating the end of the frame segment. Click on Frame 1 of the Welcome layer. Press the Delete key as shown in figure 1.4.

Notice that Frame 1 of the Welcome layer now has a white circle rather than a black circle. Also notice that the frames between 1 and 99 are now white which means they are empty as depicted in figure 1.5.



Fig. 1.4: Creating a Key frame



Fig. 1.5: Add a Key frame

2.3 Commenting Frames

Since our movie is only 10 seconds long, it's not difficult to remember what happens in the project. However if your movie were 10 minutes, there could be hundreds of Key frames. Frame comments allow developers to record text descriptions for Key frames. This helps you remember the events that take place at important points in your animation. These comments won't be exported or viewed by end users, so they don't need to be pretty, just descriptive.

To add a comment to a frame

Select Frame 100 of the Welcome layer. You should select the frame where you wish the comment to be recorded. In the Properties panel, click in the area under the word Frame.

Type: //question appears

All comments should start with 2 slashes. This indicates that they are comments, not frame labels. Notice that your frame comment now appears in the Timeline as shown in figure 1.6 and 1.7.

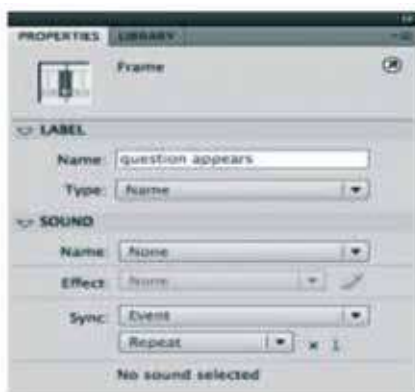


Fig. 1.6: Comment Panel

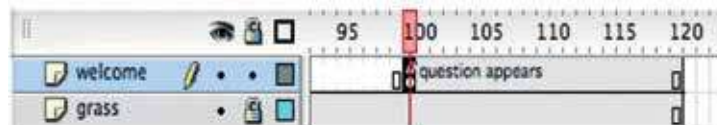


Fig. 1.7: Add a Comment

Section-2: Exercise Questions

- Q. 1: How do you insert a frame on the timeline?
- Q. 2: What is the use of frame comments?
- Q. 3: When are key frames actually added?

Q. 4: Fill in the blanks

- (a) are static and contain the same content as the preceding Key frame and are sometimes called “slaves” to the Key frame.
- (b) Most developers use a frame rate of frames per second (fps) which is the default in Flash.

Q. 5: True or False?

- (a) The frames are white when they are empty and turn gray when they contain content.
- (b) The first frame of each layer is a Key frame

Further Readings

1. <http://www.gimp.org/>
2. http://animation.about.com/od/glossaryofterms/g/layer_def.htm

Section-3: Motions and Shape Tweening

3.1 Creating Animation

Flash contains cool features for creating a variety of animations. Flash offers 3 ways to create animation.

- (i) **Frame-by-Frame:** The most flexible approach is called frame-by-frame animation. With this type, the developer makes small, individual changes to objects on the Stage within every frame. Although effective, this can be very time-consuming. If you’ve tried making a flip-book or creating Claymation, you’re familiar with this approach. Since Flash is a frame-based program, it’s easy to move from frame-to-frame to make these changes. With frame-by-frame animation, you can make incremental changes in the placement of the object to give the appearance of motion (i.e., swinging pendulum, playing on seesaw, or raft floating on water). In addition, you can change the object itself, so it looks like it had been transformed (i.e., pumpkin turns into a jack-o-lantern, a frown turns into a smile, a plant grows).
- (ii) **Tweening:** It is a technique used to make animation much easier. The developer identifies a start point and end point, then the software figures out the position of the objects “between.” Flash contains two types of tweens: motion tweens and shape tween.

- (iii) **Motion Tweening:** Motion tweens move objects from one place on the Stage to another. For example, we'll move the airplane across the Stage. The developer places an object at the starting point at a Key frame. Then, the developer creates a Key frame somewhere along the Timeline and moves an instance of the object to the end point. The creator returns to the beginning frame and chooses the motion tween option. The software then moves the object on a direct path from the beginning to the ending point. Flash provides options to tweak the tweening so that it looks more realistic. For example, motion guide can be used to move the animation along an established line so it looks like a plane is taking off. Easing is used to make it look like a ball is really bouncing.
- (iv) **Shape Tweening:** Shape tweens morph one image into another image. For example, we'll change the shape of the cloud as it moves across the Stage. You can also change circles into stars, a seed into a plant, or a bike into a car. Keep in mind that shape tweening only really works well with the vector graphics. In other words, Flash shape tweening won't work well on a photograph of a human face, but it would work for a line-based smiley face.
- (v) **ActionScript:** Besides frame-by-frame and tweening, objects can be animated using ActionScript. In addition to animation, scripting is also used for highly interactive projects such as video games and simulations. Finally, you can also choose from pre-built animations provided by Flash.

3.2 Creating Frame-by-Frame Animation

Let's have some fun with frame-by-frame animation. For this assignment, let's create a web page banner rather than a standard screen size document. We'll begin by creating a new document and setting the page properties.

To set the page properties

Create a new document.

In the PROPERTIES area of the Properties panel, choose to Edit... next to the Size option. Change the size to 550 x 100 pixels.

Notice that you can change the Background Colour and Frame rate here as shown in figure 1.8.

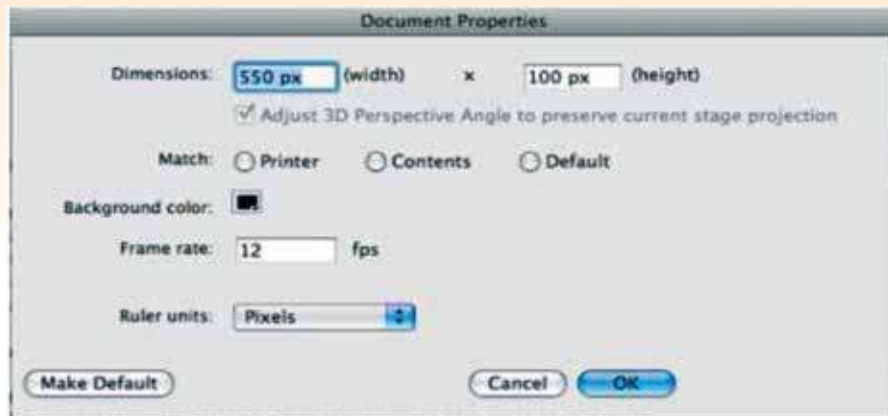


Fig. 1.8: Page Properties

Next, we'll create a picture for the simple animation. Use the line and shape tools to draw a simple stick person on the extreme left side of the screen. You can also draw a globe, basketball, car, or other object. For a circle, click the Rectangle tool and draw a circle. Use the line tool to create lines. Building the animation involves creating a series of Key frames with small changes to the figure. The default setting is 12 frames per second. Let's create movement every 5-10 frames.

To create frame-by-frame animation

Click on Frame 5. Pull down the Insert menu, choose Timeline and Key frame. Note the dot. Click on Frame 15. Add another Key frame. Also add Key frames on 20, 25, 30, and 40. Click on Frame 1. Pull down the Edit menu, Select All. Pull down the Edit menu, choose Copy. Go to Frame 5. Pull down the Edit menu, choose Paste. Go to all the other frames and paste. Go back and make small changes on each Key frame such as moving legs or arms. You can also go to the Modify menu and choose Transform to make changes such as rotation. This process is depicted in figure 1.9.

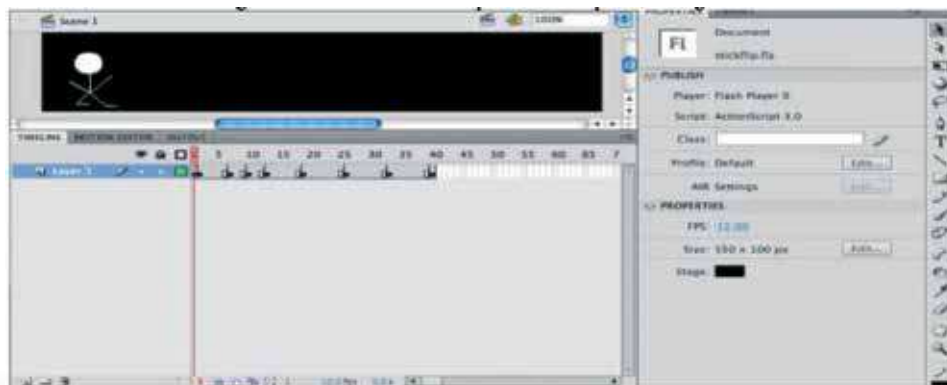


Fig. 1.9: Creating Frame by Frame Animation

3.3 Creating Motion Tween Animation

The most common animation technique is motion tweening. You simply provide Flash with a starting point on a Key frame and an ending point on another Key frame. Then, you set the motion “tween” and watch it animation!

The motion tween animation options have changed dramatically from earlier versions of Adobe Flash. If you have an older version, you’ll need to adjust the instructions. The newest version contains both the “classic motion tween” as well as a new approach. We’ll try both in this section.

Classic Motion Tween

Keep in mind that you can only use graphic symbols and movie clips as motion tweens. We’ll create a motion tween to move our airplane across the Stage.

To create a “classic” motion tween for airplane

Unlock the airplane layer if you locked it earlier. Ensure that your object is a SYMBOL before you create your animation. If you’re not sure, pull down the Modify Toolbars, menu and select Convert to Symbol. Give it a name and make it a graphic.

Select Frame 120 of the airplane layer. Pull down the Insert menu, choose Timeline, select Key frame. Notice that the airplane copied to your new Key frame. Drag the airplane so it appears in the upper right corner of the Stage. Sequence of frame 1 and 2 is shown in figure 1.10 and 1.11.



Fig. 1.10: MT Key Frame-1



Fig. 1.11: MT Key Frame-2

Motion Tween

The second option for tween animation is the new Motion Tween option. The advantage of this approach is that you don’t need to mess with creating Key frames, the system creates a guide automatically.

To create a motion tween for our airplane (as shown in figure 1.12):

Unlock the airplane layer if you locked it earlier. Ensure that your object is a SYMBOL before you create your animation. If you're not sure, pull down the ModifToolbars, y menu and select Convert to Symbol. Give it a name and make it a graphic. Click on Frame 1 of the airplane layer. Pull down the Insert menu, select Motion Tween.

The Timeline will turn blue from Frame 1 to 120. Click on the Frame where you'd like to see your first movement, this case Frame 10. Drag the airplane to the new location an inch above the runway. A special Key frame called a Property Key frame will be added to the Timeline after you've changed the position of the graphic. It looks like a filled triangle. Click on Frame 20. Drag the airplane back down to the runway.

Make movements for Frames 40, 80, 100, and 120 to simulate the way an airplane bounces down the runway.

Remember that you can start and end an animation off the stage if you wish.

If you wish to adjust the guide, click on the Property Key frame containing one of the Key frames made by the tween and move the graphic. The guide line will automatically be adjusted.

The new Motion Editor provides additional tools for enhancing your Motion Tween. Click the MOTION EDITOR tab next to the TIMELINE tab. This allows you to change No Ease to Simple (Slow). You also have more precise movement of your Property Key frames.

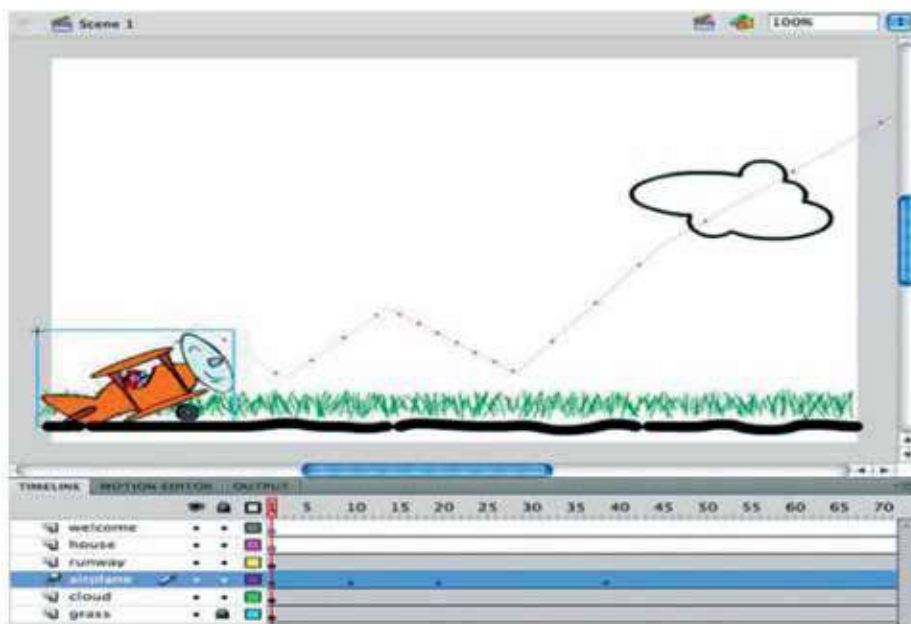


Fig. 1.12: The New Motion Tween

3.4 Creating Shape Tween Animation

It's fun to morph images. Tweening shapes involves changing an object's shape at one or more points in the animation. The software creates the in-between shapes for you. You can combine changes in position as well as changes in size, colour, and transparency. You'll want to work with just one shape per layer.

Like motion tweening that uses graphic symbols, you can't shape tween on a symbol, text, or a group unless you break them apart. Shape tweening works best on objects that you create using the draw tools such as our cloud. Let's try a couple shape tweens. First, we'll create a simple alphabet book. We'll start with an apple that changes into the letter A.

To create the shape tween

Create a new document. Click in Frame 1 of Layer 1. Choose the paintbrush and the colour red from the paint bucket. Draw an apple in the lower left corner of the screen.

Select it. Pull down the Modify menu and choose Convert to Symbol, Graphic.

Now it's in the Library in case you need it later. Select your apple and pull down the Modify menu and select Break Apart. This will turn your apple into tiny pieces.

Go Frame 20 and Insert a Key frame. Click on this Frame. Use your paintbrush to draw a letter A on the right side of the screen. Select your A. Pull down the Modify menu and choose Convert to Symbol, Graphic. Now it's in the Library in case you need it later.

Select your A and pull down the Modify menu and select Break Apart. This will turn your A into tiny pieces. Go back to Frame 1. Look below the Stage for the Properties window.

Choose Shape from the Tween drop-down menu. Scrub the Timeline to see your results!

Let's return to the airplane example. Rather than the cloud moving left to right, we want it to move from right to left. This movement will give us the illusion that the plane is passing the clouds. To create a shape tween using our cloud as shown in figure 1.13.

Select Frame 120 of the cloud layer. Actually, you could select any frame you want as the end of your morph process. Pull down the Insert menu, choose Timeline, select Key frame. A Key frame will appear on the Timeline. Unlock the cloud layer.



Fig. 1.13: Shape Tween Animation for Alphabet Book

Section-3: Exercise Questions

- Q.1: What is tweening?
- Q.2: Name 2 types of tweens and state them briefly?
- Q. 3: Why is action script used?
- Q. 4: How do you set the frame properties?
- Q. 5: Explain the method to create a “classic” motion tween for airplane’s example?
- Q. 6: How do you create the shape tween?
- Q. 7: Fill in the blanks
- (a) move objects from one place on the Stage to another
 - (b) morph one image into another image
- Q. 8: State True or False
- (a) Flash shape tweening won’t work well on a photograph of a human face, but it would work for a linebased smiley face.
 - (b) You can only use graphic symbols and movie clips as motion tweens.
 - (c) The software creates the in-between shapes for you and you can combine changes in position as well as changes in size, color, and transparency.

Further Readings

1. http://www.gimp.org/tutorials/Simple_Animations/
2. <http://gimpmagazine.org/>
3. <http://animation.about.com/od/flashanimationtutorials/f/Whats-The-Difference-Between-Shape-Motion-And-Classic-Tweens.htm>

Section-4: Importing AV file

Audio and video are working as ear and eye of multimedia. Both of them are heavily contributing to any multimedia application. Let us discuss something about the association for these field with multimedia.

4.1 Sound and Audio

Sound is mechanical energy disturbances that propagates through matter as wave. Various properties of sound are as follows;

- ◆ Frequency
- ◆ Wave length
- ◆ Period
- ◆ Amplitude
- ◆ Velocity or speed

Noise and sound are often sense the same. The noise is an unwanted and undesirable part of a signal. Human being perceive sound by the sense of hearing. Sound i.e created through vibration always travel through air and can be heard by human, and vibrations that travel through all forms of matter, gases and solids. The matter that supports sound is called the medium.

The scientific study of sound is called acoustics. The sound portion of a program or a track recorded on a videotape which contains sound, music or narration is called Audio.

4.2 Audio File Formats

An audio file format is a container format for storing audio data on a computer system. There are numerous file formats for storing audio files. It is important to distinguish between a file format and codec though most audio file formats support only one audio codec, a file format may support multiple code.

- ◆ Common format such as WAV, AIFF, and AU
- ◆ Formats as lossless compression such as FLAC, MONKEYS AUDIO (file name extension APE) Wav pack, Shorten,
- ◆ TTA, apple Lossless and Lossless Window Media Audio (WMA)
- ◆ Formats with compression such as MP3, Vorbis, WMA and AAC.

Uncompressed /common audio format

There is one major uncompressed audio format: PC. It is usually stored as .wav on windows. WAV is a flexible file format designed to store more less any combination of sampling rates arbitrates. The WAV format is based on the RIFF file format, which is similar to the IFF format.

Lossless audio formats

Lossless audio format (such as TTA and FLAC) provide a compression ratio of about 2:1 sometimes more. *A lossless compressed format would require more processing for the same time recoded, but would be more efficient in terms of space used. WAV, like any other uncompressed format encodes all sounds, whether they are complex sounds or absolutes silence, with the same number of bits per unit of time.*

It is in the area of hardware support that FLAC is ahead of the competition. FLAC is supported by a wide variety of portable audio playback devices.

4.3 Lossy Audio Formats

These file formats are based on sound type that remove audio rate that human cannot or can hardly hear, e.g.a low volume sound after a big volume sound. MP3 and Vorbis are good examples. One of the popular lossy file format is MP3 , which uses the MPEG-I audio layer 3 code to provide acceptable loosy compression for music files. The compression is about 10:1 as compared to uncompressed WAV files. A CD with MP3 files can store upto 11 hours of music , compared to 74 minute of the standard cdda, which uses uncompressed PCM.Vorbis is an unpatented free codec.

Multiple channel

Nowadays movie theaters have upgraded their sound systems to surround sound systems that carry more than two channels. Mostly used are Advanced Audio Coding or AAC (used by Apple's i tunes) and DOLBY Digital, also known as AC-3. Both codec are copyrighted and encoders/decoders cannot be offered without paying a license fee.

Digital video (DV)

Digital video is a type of video recording system that use digital recording system. this generic term is not to be confused with the generic term DV which is specific type of digital video. Digital video can be characterized by few variables :

- ◆ Frame rate: The number of frames displayed per second. The illusion of motion can be experienced as 12 frames per second.
- ◆ Frame dimension: The width and height of the image can be expressed in the number of pixel.
- ◆ Pixel depth: The number of bits per pixel.

4.4 Type of video file format

Captured DV is stored in computer system in AVI file. Which is Microsoft's standard file format for video files as shown in figure 1.14.

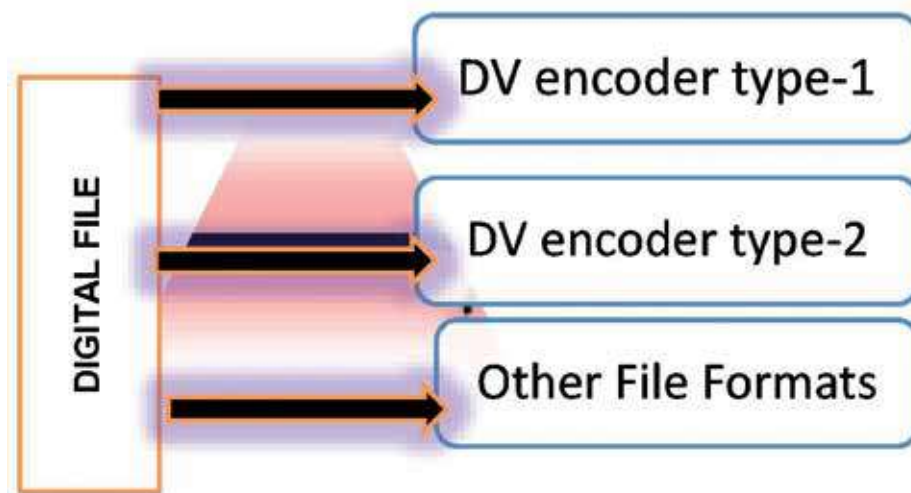


Fig. 1.14

DV encoder type-1: This format is fully supported by direct show with an Encoder Type-1 AVI file where the raw DV interleaved data stream is simply written into the file.

DV encoder type-2: Encoder -Type 2 produces a VFW compatible AVI file format. The advantage is that it can be read by the older applications that do not support the file format

Other File Formats: Those are used for storage and viewing of video by and on computer systems (with the exception of MPEG formats). These are mentioned below:

- ◆ **MPEG-1** : (Moving Picture Expert Group format-1) is an industry standard encoding format that is widely used in industry.
- ◆ **MPEG-2**: It is of higher quality than the first one. but the file sizes are 3 to 4 times larger than MPEG-1
- ◆ **QUICKTIME**: It is the video format devised by and used by Apple and can be used at varying quality and file sizes.
- ◆ **REAL VIDEO**: It is a streaming video format used for distributing video in real-time over the internet.

Section-4: Exercise Questions

- Q.1: What are the various properties of sound?
- Q. 2: Name one major uncompressed audio format?
- Q. 3: Captured DV is stored in computer system in?
- Q. 4: Which is the video format devised by and used by Apple that can be used at varying quality and File sizes?
- Q. 5: Explain DV encoder type-1?
- Q. 6: Why is DV encoder type-2 used?
- Q. 7: Explain “real video”?
- Q. 8: Explain the concept of digital video?
- Q. 9: Fill in the blanks
- (a) The scientific study of sound is called
- (b) produces a VFW compatible AVI file format.

Further Readings

1. <http://www.gimp.org/>
2. <http://graphicssoft.about.com/od/gimptutorials/a/Supported-File-Types.htm>

Section-5: Multimedia Publishing

Multimedia publishing is presenting multimedia works that have been authored and produced by a multimedia artist or video editor. Multimedia authoring and production, that start the processes from source materials and developed (designed) to a formal work of art. Multimedia publishing is outputting the final work to either print format or digital format on discs or on the web.

Publishing software: Various type of publishing software

Adobe premier

The premier publisher of multimedia software. Products include Adobe Director (multimedia authoring), Dreamweaver (web publishing), Photoshop (photo editing). (more.store.adobe.com)

Create Together

Multimedia environment that integrates creation, collaboration, communication, problem solving, and publishing in one seamless tool.

Create animated games, randomly generated puzzles, interactive simulations, searchable multimedia databases, hyperlinked presentations and more.

Software products for producing interactive multimedia presentations, training courses, product catalogues and kiosk systems. (www.dazzlersoft.com)

Flying Popcorn

New computer users can create multimedia contents with 3D sound effects, images, geometrical shapes, text and movie files. Also, professional users will find abundant application potentials and dazzling transformation capabilities to develop sophisticated contents. (www.parasys.net)

GraFX Saver Pro

Create and distribute your own professional screen savers with over 30 image, video, and audio formats. (www.grafx saver.com)

Hyper Studio

Classroom multimedia authoring program. (www.hyperstudio.com)

Live Stage

Quicktime authoring application with the ability to converge more than 200 media types. (www.totallyhip.com)

Media Blender

Easy to use multimedia authoring package. Access MediaBlender from any computer with Internet access, host the software on your own servers for better performance, and load a stand-alone application on your computer for use without an Internet connection. (www.tech4learning.com)

Mistyk Media

All-in-one software for conversion, editing, recording, burning, playback, and more for nearly all audio, video, and graphics formats.

(www.mystikmedia.com)

Movie Works

Combines simple-to-use video, sound, animation, paint and image editing tools with a powerful time-based, object-oriented sequencing and authoring program. The five integrated MovieWorks Deluxe production applications work together seamlessly, making it easy to incorporate analog or digital video, buttons, photos, graphics, animations, 3D, narration, MIDI, MP3, CD music, text and titling, and even virtual reality movies into stunning, professional-quality multimedia productions.

GIMP as publishing software

It provides a full set of word-processing features as well as fine control over placement of text graphics, so that you can create newsletter, advertisement, books, and other type of documents.

GIMP is an image editing piece of software. Before we get into a lot of details about the software the important things are.

1. The software is available not only for Windows and Apple operating systems but for Linux also. This means that no system will be left out of the party and you will have complete cross system compatibility for your files.
2. You can get this software online as a free download. That is right - you will not be charged for the software.

Section-5: Exercise Questions

- Q. 1: What do you understand by multimedia publishing?
- Q. 2: State five types of multimedia publishing software?
- Q. 3: Explain GIMP as publishing software?

Further Readings

1. <http://www.gimp.org/>
2. <http://graphicssoft.about.com/od/gimptutorials/a/Supported-File-Types.htm>

Books on GIMP software

1. Niveles y Curvas con GIMP
2. The Book of GIMP
3. GIMP for Absolute Beginners
4. Beginning GIMP: From Novice to Professional, Second Edition
5. GIMP for Textile Design
6. Essential GIMP for Web Professionals
7. GIMP - The Official Handbook

2

DIGITAL CONTENT CREATION – ADDING STYLES TO WEB PAGES(CSS)

Learning Objectives

Section-1: Basics About HTML

Section-2: Basics About CSS

Section-3: CSS Properties — border, box, font, margin, padding CSS classes

Section-1 Basics About HTML

1.1 HTML Basics

This Chapter leads you to the basics of Hyper Text Mark up Language (HTML). HTML is the building block for web pages. You will learn to use HTML to author an HTML page to display in a web browser.

Objectives

By the end of this Chapter, you will be able to:

- ◆ Use a text editor to author an HTML document.
- ◆ Be able to use basic tags to denote paragraphs, emphasis or special type.
- ◆ Create hyperlinks to other documents.
- ◆ Create an email link.
- ◆ Add images to your document.
- ◆ Use a table for layout.

Prerequisites

You will need a text editor, such as Notepad and an Internet browser, such as Internet Explorer or Netscape Navigator.

Q: What is Notepad and where do I get it?

A: Notepad is the default Windows text editor. On most Windows systems, click your Start button and choose Programs then Accessories. Its icon is a little blue notebook.

Mac Users: SimpleText is the default text editor on the Mac. In OSX use TextEdit and change the following preferences: Select (in the preferences window) Plain text instead of Rich text and then select Ignore rich text commands in HTML files. This is very important because if you don't do this HTML codes probably won't work.

One thing you should avoid using is a word processor (like Microsoft Word) for authoring your HTML documents.

What is an html File?

The documents themselves are plain text files with special “tags” or codes that a web browser uses to interpret and display information on your computer screen.

- ◆ HTML stands for Hyper Text Markup Language
- ◆ An HTML file is a text file containing small markup tags
- ◆ The markup tags tell the Web browser how to display the page
- ◆ An HTML file must have an htm or html file extension

Try It

Open your text editor and type the following text:

```
<html>
<head>
<title>My First Webpage</title>
</head>
<body>
This is my first homepage. <b>This text is bold</b>
</body>
</html>
```

Save the file as **mypage.html**. Start your Internet browser. Select **Open** (or Open Page) in the **File** menu of your browser. A dialog box will appear. Select **Browse** (or Choose File) and locate the html file you just created - **mypage.html** - select it and click **Open**. Now you should see an address in the dialog box, for example **C:\MyDocuments\mypage.html**. Click **OK**, and the browser will display the page.

Example Explained

What you just made is a skeleton html document. This is the minimum required information for a web document and all web documents should contain these basic components.

The first tag in your html document is `<html>`. This tag tells your browser that this is the start of an html document.

The last tag in your document is `</html>`. This tag tells your browser that this is the end of the html document.

The text between the `<head>` tag and the `</head>` tag is header information.

Header information is not displayed in the browser window.

The text between the `<title>` tags is the title of your document.

The `<title>` tag is used to uniquely identify each document and is also displayed in the title bar of the browser window.

The text between the `<body>` tags is the text that will be displayed in your browser.

The text between the `<b>` and `</b>` tags will be displayed in a bold font.

1.2 Basic HTML Tags

The most important tags in HTML are tags that define headings, paragraphs and line breaks.

Basic HTML Tags

Tag	Description
<code><html></code>	Defines an HTML document
<code><body></code>	Defines an document's body
<code><h1></code> to <code><h6></code>	Defines header 1 to header 6
<code><p></code>	Defines a paragraph
<code>
</code>	Inserts a single line break
<code><hr></code>	Defines a horizontal rule
<code><!--></code>	Defines a comment

Headings

Headings are defined with the <h1> to <h6> tags. <h1> defines the largest heading while <h6> defines the smallest.

```
<h1>This is a heading</h1>
<h2>This is a heading</h2>
<h3>This is a heading</h3>
<h4>This is a heading</h4>
<h5>This is a heading</h5>
<h6> This is a heading</h6>
```

HTML automatically adds an extra blank line before and after a heading. A useful heading attribute is align.

```
<h5 align="left">I can align headings </h5>
<h5 align="center">This is a cantered heading </h5>
<h5 align="right">This is a heading aligned to the right </h5>
```

Paragraphs

Paragraphs are defined with the <p> tag. Think of a paragraph as a block of text. You can use the align attribute with a paragraph tag as well.

```
<p align="left">This is a paragraph</p>
<p align="center">this is another paragraph</p>
```

Line Breaks

The
 tag is used when you want to start a new line, but don't want to start a new paragraph. The
 tag forces a line break wherever you place it. It is similar to single spacing in a document.

This Code	Would Display
This is a para graph with line breaks	This is a para graph with line breaks

Note:- The
 tag has no closing tag.

Horizontal Rule

The `<hr>` element is used for horizontal rules that act as dividers between sections, like this:

The horizontal rule does not have a closing tag. It takes attributes such as `align` and `width`. For instance:

This Code	Would Display
<code><hr width="50%" align="center"></code>	

Comments in HTML

The comment tag is used to insert a comment in the HTML source code. A comment can be placed anywhere in the document and the browser will ignore everything inside the brackets. You can use comments to write notes to yourself, or write a helpful message to someone looking at your source code.

This Code	Would Display
<code><p> This html comment would <!-- This is a comment --> be displayed like this.</p></code>	This HTML comment would be displayed like this.

Note:- You don't see the text between the tags `<!--` and `-->`. If you look at the source code, you would see the comment. To view the source code for this page, in your browser window, select **View** and then select **Source**.

Other HTML Tags

There are logical styles that describe what the text should be and physical styles which actually provide physical formatting. It is recommended to use the logical tags and use style sheets to style the text in those tags.

Logical Tags:		Physical Tags:	
Tag	Description	Tag	Description
<code><abbr></code>	Defines an abbreviation	<code></code>	Defines bold text
<code><acronym></code>	Defines an acronym	<code><big></code>	Defines big text
<code><address></code>	Defines an address element	<code><i></code>	Defines <i>Italic</i> text
<code><cite></code>	Defines citation	<code><small></code>	Defines text
<code><code></code>	Defines computer code text	<code><sup></code>	Defines ^{superscripted} text

Logical Tags:		Physical Tags:	
Tag	Description	Tag	Description
<blockquote>	Defines a long quotation	<sub>	Defines _{subscripted} text
	Defines text	<tt>	Defines teletype text
<dfn>	Defines a definition term	<u>	Deprecated. Use styles instead
	Defines emphasized text		
<ins>	Defines inserted text		
<kbd>	Defines keyboard text		
<pre>	Defines preformatted text		
<q>	Defines short quotation		
<samp>	Defines sample computer code		
	Defines strong text		
<var>	Defines a variable		

Character tags like and produce the same physical display as and <i> but are more uniformly supported across different browsers.

1.3 HTML Character Entities

Some characters have a special meaning in HTML, like the less than sign (<) that defines the start of an HTML tag. If we want the browser to actually display these characters we must insert character entities in place of the actual characters themselves.

The Most Common Character Entities:

Result	Description	Entity Name	Entity Number
	Non breaking space	 	
<	Less than	<	<
>	Greater than	>	>
&	Ampersand	&	&
“	Quotation mark	"	"
‘	apostrophe	' (does not work in IE)	'

A character entity has three parts: an ampersand (&), an entity name or an entity number, and finally a semicolon (;). The & means we are beginning a special character, the ; means ending a special character and the letters in between are sort of an abbreviation for what it's for. To display a less than sign in an HTML document we must write: **<** or

< The advantage of using a name instead of a number is that a name is easier to remember. The disadvantage is that not all browsers support the newest entity names, while the support for entity numbers is very good in almost all browsers.

Note: Entities are case sensitive.

1.4 HTML Background

Background

The `<body>` tag has two attributes where you can specify backgrounds. The background can be a color or an image.

Bgcolor

The `bgcolor` attribute specifies a background-color for an HTML page. The value of this attribute can be a hexadecimal number, an RGB value, or a color name:

```
<body bgcolor="#000000"> <body bgcolor="rgb(0,0,0)"> <body bgcolor="black">
```

The lines above all set the background-color to black.

Background

The `background` attribute can also specify a background-image for an HTML page. The value of this attribute is the URL of the image you want to use. If the image is smaller than the browser window, the image will repeat itself until it fills the entire browser window.

```
<body background="clouds.gif"> <body  
background="http://profdevtrain.austincc.edu/html/graphics/clouds.gif">
```

The URL can be relative (as in the first line above) or absolute (as in the second line above).

If you want to use a background image, you should keep in mind:

- ◆ Will the background image increase the loading time too much?
- ◆ Will the background image look good with other images on the page?
- ◆ Will the background image look good with the text colors on the page?
- ◆ Will the background image look good when it is repeated on the page?
- ◆ Will the background image take away the focus from the text?

Note: The `bgcolor`, `background`, and the `text` attributes in the `<body>` tag are deprecated in the latest versions of HTML (HTML 4 and XHTML).

The World Wide Web Consortium (W3C) has removed these attributes from its recommendations. Cascading Style sheets (CSS) should be used instead (to define the layout and display properties of HTML elements).

Try It Out!

Open your text editor and type the following text:

```
<html>
<head>
<title>My First Webpage</title>
</head>
<body background="http://profdevtrain.austincc.edu/html/graphics/clouds.gif" bgcolor
="#EDDD9E">
<h1 align="center">My First Webpage</h1>
<p>Welcome to my <strong>first</strong> webpage. I am writing this page using a
text editor and plain old html.</p>
<p>By learning html, I'll be able to create webpages like a <del>beginner</del>
pro....<br>
which I am of course.</p>
</body>
</html>
```

Save your page as **mypage3.html** and view it in your browser.

1.5 HTML Links

HTML uses the <a> anchor tag to create a link to another document or web page.

The Anchor Tag and the Href Attribute

An anchor can point to any resource on the Web: an HTML page, an image, a sound file, a movie, etc. The syntax of creating an anchor: Text to be displayed

The <a> tag is used to create an anchor to link from, the href attribute is used to tell the address of the document or page we are linking to, and the words between the open and close of the anchor tag will be displayed as a hyperlink.

This Code	Would Display
Visit ACC!	Visit ACC!

The Target Attribute

With the target attribute, you can define where the linked document will be opened. By default, the link will open in the current window. The code below will open the document in a new browser window:

```
<a href=http://www.austincc.edu/ target="_blank">Visit ACC!</a>
```

Email Links

To create an email link, you will use mailto: plus your email address. Here is a link to ACC's Help Desk:

```
<a href="mailto:helpdesk@austincc.edu">Email Help Desk</a>
```

To add a subject for the email message, you would add ?subject= after the email address. For example:

```
<a href="mailto:helpdesk@austincc.edu?subject=Email Assistance">Email Help Desk</a>
```

1.6 HTML Images

The Image Tag and the Src Attribute

The tag is empty, which means that it contains attributes only and it has no closing tag. To display an image on a page, you need to use the src attribute. Src stands for "source". The value of the src attribute is the URL of the image you want to display on your page. The syntax of defining an image:

This Code	Would Display
	

Image Dimensions

When you have an image, the browser usually figures out how big the image is all by itself. If you put in the image dimensions in pixels however, the browser simply reserves a space for the image, then loads the rest of the page. Once the entire page is loaded it can go back and fill in the images. Without dimensions, when it runs into an image, the browser has to pause loading the page, load the image, then continue loading the page. The chef image would then be:

Open the file **mypage2.html** in your text editor and add code highlighted in bold:

```
<html>
<head>
<title>My First Webpage</title>
</head>
<body>
<h1 align="center">My First Web page</h1>
<p>Welcome to my first webpage. I am writing this page using a text editor and plain
old html.</p>
<p>By learning html, I'll be able to create web pages like a pro....<br>
which I am of course.</p>
<!-- Who would have guessed how easy this would be :) -->
<p></p>
<p align="center">This is my Chef</p>
</body>
</html>
```

1.7 Tables

Tables are defined with the <table> tag. A table is divided into rows (with the <tr> tag), and each row is divided into data cells (with the <td> tag). The letters td stands for table data, which is the content of a data cell. A data cell can contain text, images, lists, paragraphs, forms, horizontal rules, tables, etc.

This Code	Would Display				
<pre><table> <tr> <td>row 1, cell 1</td> <td>row 1, cell 2</td> </tr> <tr> <td>row 2, cell 1</td> <td>row 2, cell 2</td> </tr> </table></pre>	<table><tr><td></td><td></td></tr><tr><td></td><td></td></tr></table>				

Tables and the Border Attribute

To display a table with borders, you will use the border attribute.

This Code	Would Display	
<pre><table border="1"> <tr> <td>Row 1, cell 1</td> <td>Row 1, cell 2</td> </tr> </table></pre>		

Summary

HTML is the markup language for building Web pages and traditionally has combined physical and logical structuring ideas. Elements—in the form of tags such as **** and ****—are embedded within text documents to indicate to browsers how to render pages.

The rules for HTML are fairly simple and compliance can be checked with a process called validation. Unfortunately, these rules have not been enforced by browsers in the past. Because of this looseness, there has been a great deal of misunderstanding about the purpose of HTML, and a good portion of the documents on the Web do not conform to any particular official specification of HTML. Stricter forms of HTML, and especially the introduction of XHTML, attempt to impose a more rigid syntax, encourage logical markup, and leave presentational duties to other technologies such as Cascading Style Sheets. While very widespread, use of strict markup has yet to occur on the Web. Web developers should aim to meet standards to future-proof their documents and more easily address the various issues that will certainly arise while getting browsers to render them properly.

Section-1: Exercise Questions

- Q. 1: What are the pre requisites to learn the basics of HTML?
- Q. 2: How do you define a HTML file?
- Q. 3: What is the tag for making a comment?
- Q. 4: What dose <cite> and <pre> tag mean?
- Q. 5: What is the syntax to define background color in HTML?

- Q. 6: Explain the procedure to view source code in Mozilla Firefox, Google Chrome & Internet Explorer?
- Q. 7: List out any three popular web browsers?
- Q. 8: What do you mean by an anchor tag?
- Q. 9: How do you insert an image and create a table in the HTML page?
- Q. 10: What is the tag to insert a single line break?
- Q. 11: Fill in the blanks:
- (a) Word processors like should be avoided.
 - (b) The tells the web browser how to display the page
 - (c) is the tag used for defining the Sample computer code
 - (d) Acronym for HTTP
 - (e) Acronym for HTML
- Q. 10: What does W3C stand for?

Section-2: Basics About CSS

Cascading Style Sheets (CSS) describe how documents are presented on screens, in print, or perhaps how they are pronounced. W3C has actively promoted the use of style sheets on the Web since the Consortium was founded in 1994. Cascading Style Sheets (CSS) provide easy and effective alternatives to specify various attributes for the HTML tags. Using CSS, you can specify a number of style properties for a given HTML element. Each property has a name and a value, separated by a colon (:). Each property declaration is separated by a semi-colon (;).

2.1 Advantages of CSS

- ◆ **CSS saves time** – You can write CSS once and then reuse same sheet in multiple HTML pages. You can define a style for each HTML element and apply it to as many Web pages as you want.
- ◆ **Pages load faster** – If you are using CSS, you do not need to write HTML tag attributes every time. Just write one CSS rule of a tag and apply to all the occurrences of that tag. So less code means faster download times.
- ◆ **Easy maintenance** – To make a global change, simply change the style, and all elements in all the web pages will be updated automatically.

- ◆ **Superior styles to HTML** – CSS has a much wider array of attributes than HTML so you can give far better look to your HTML page in comparison of HTML attributes.
- ◆ **Multiple Device Compatibility** – Style sheets allow content to be optimized for more than one type of device. By using the same HTML document, different versions of a website can be presented for handheld devices such as PDAs and cell phones or for printing.
- ◆ **Global web standards** – Now HTML attributes are being deprecated and it is being recommended to use CSS. So it's a good idea to start using CSS in all the HTML pages to make them compatible to future browsers.

Example

First let's consider an example of HTML document which makes use of tag and associated attributes to specify text color and font size:

```
<!DOCTYPE html>
<html>
<head>
<title>HTML CSS</title>
</head>
<body>
<p><font color="green"size="5">Hello, world!</font></p>
</body>
</html>
```

We can re-write above example with the help of Style Sheet as follows:

```
<!DOCTYPE html>
<html>
<head>
<title>HTML CSS</title>
</head>
<body>
<p style="color:green;font-size:24px;">Hello, World!</p>
</body>
</html>
```

This will produce following result:

Hello, World!

You can use CSS in three ways in your HTML document:

- ◆ **External Style Sheet** – Define style sheet rules in a separate .css file and then include that file in your HTML document using HTML <link> tag.
- ◆ **Internal Style Sheet** – Define style sheet rules in header section of the HTML document using <style> tag.
- ◆ **Inline Style Sheet** – Define style sheet rules directly along with the HTML elements using style attribute. Let's see all the three cases one by one with the help of suitable examples.

2.2 External Style Sheet

If you need to use your style sheet to various pages, then it's always recommended to define a common style sheet in a separate file. A cascading style sheet file will have extension as .css and it will be included in HTML files using <link> tag.

Example

Consider we define a style sheet file style.css which has following rules:

```
.red{  
color: red;  
}  
.thick{  
font-size:20px;  
}  
.green{  
color:green;  
}
```

Here we defined three CSS rules which will be applicable to three different classes defined for the HTML tags. I suggest you should not bother about how these rules are being defined because you will learn them while studying CSS.

```
<!DOCTYPE html>
<html>
<head>
<title>HTML External CSS</title>
<link rel="stylesheet" type="text/css" href="/html/stml/style.css">
/head>
<body>
<p class="red">This is red</p>
<p class="thick">This is thick</p>
<p class="green">This is green</p>
<p class="thick green">This is thick and green</p>
</body>
</html>
```

This will produce following result:

This is red

This is thick

This is green

This is thick and green

2.3 Internal Style Sheet

If you want to apply Style Sheet rules to a single document only then you can include those rules in header section of the HTML document using <style> tag.

Rules defined in internal style sheet override the rules defined in an external CSS file.

Example

Let's re-write above example once again, but here we will write style sheet rules in the same HTML document using <style> tag:

```
<!DOCTYPE html>
<html>
<head>
<title>HTML Internal CSS</title>
<style type="text/css">
.red{
color: red;
}
.thick{
font-size:20px;
}
.green{
color:green;
}
</style>
</head>
<body>
<p class="red">This is red</p>
<p class="thick">This is thick</p>
<p class="green">This is green</p>
<p class="thick green">This is thick and green</p>
</body>
</html>
```

This will produce following result:

This is red

This is thick

This is green

This is thick and green

2.4 Inline Style Sheet

You can apply style sheet rules directly to any HTML element using style attribute of the relevant tag. This should be done only when you are interested to make a particular change in any HTML element only.

Rules defined inline with the element overrides the rules defined in an external CSS file as well as the rules defined in <style> element.

Example

Let's re-write above example once again, but here we will write style sheet rules along with the HTML elements using style attribute of those elements.

```
<!DOCTYPE html>
<html>
<head>
<title>HTML Inline CSS</title>
</head>
<body>
<p style="color:red;">This is red</p>
<p style="font-size:20px;">This is thick</p>
<p style="color:green;">This is green</p>
<p style="color:green;font-size:20px;">This is thick and green</p>
</body>
</html>
```

This will produce following result:

This is red

This is thick

This is green

This is thick and green

Section-2: Exercise Questions

- Q.1: Why do we use CSS? What are its basic advantages?
- Q.2: What are the ways in which you can use CSS in your HTML document?
- Q.3: What are inline style sheets?
- Q.4: How are internal style sheets used?
- Q.5: Does the use of CSS saves time?
- Q.6: Fill in the blanks:
- (a) Defines style sheet rules in a separate .css file and then include that file in your HTML document using HTML <link> tag.
 - (b) The property is used to create a small-caps effect
 - (c) A cascading style sheet file will have extension as and it will be included in HTML files using tag.
 - (d) Each property declaration is separated by a
- Q. 7: State True or False?
- (a) Style sheets allow content to be optimized for more than one type of device.
 - (b) Rules defined in internal style sheet overrides the rules defined in an external CSS file.
- Q. 8: How are external style sheets used?
- Q. 9: State difference between inline, internal & external style sheets?

Section-3: CSS Properties — border, box, font, margin, padding CSS classes

3.1 CSS Syntax – Selectors

A CSS comprises of style rules that are interpreted by the browser and then applied to the corresponding elements in your document. A style rule is made of three parts:

- ◆ **Selector:** A selector is an HTML tag at which style will be applied. This could be any tag like <h1> or <table> etc.
- ◆ **Property:** A property is a type of attribute of HTML tag. Put simply, all the HTML attributes are converted into CSS properties. They could be color or border etc.
- ◆ **Value:** Values are assigned to properties. For example color property can have value either red or #F1F1F1 etc.

You can put CSS Style Rule Syntax as follows:

```
selector { property: value }
```

Example: You can define a table border as follows:

```
table{ border :1px solid #C00; }
```

Here table is a selector and border is a property and given value 1px solid #C00 is the value of that property. You can define selectors in various simple ways based on your comfort. Let me put these selectors one by one.

The Type Selectors

This is the same selector we have seen above. Again one more example to give a color to all level 1 headings:

```
h1 {  
color: #36CFFF;  
}
```

The Universal Selectors

Rather than selecting elements of a specific type, the universal selector quite simply matches the name of any element type :

```
{  
color: #000000;  
}
```

This rule renders the content of every element in our document in black.

The Descendant Selectors

Suppose you want to apply a style rule to a particular element only when it lies inside a particular element. As given in the following example, style rule will apply to element only when it lies inside tag.

```
ul em {  
color: #000000;  
}
```

The Class Selectors

You can define style rules based on the class attribute of the elements. All the elements having that class will be formatted according to the defined rule.

```
.black {  
color: #000000;  
}
```

This rule renders the content in black for every element with class attribute set to black in our document. You can make it a bit more particular. For example:

```
h1.black {  
color: #000000;}
```

The ID Selectors

You can define style rules based on the id attribute of the elements. All the elements having that id will be formatted according to the defined rule.

```
#black {  
color: #000000;  
}
```

The Child Selectors

You have seen descendant selectors. There is one more type of selectors which is very similar to descendants but have different functionality. Consider the following example:

```
body > p {  
  color: #000000;  
}
```

This rule will render all the paragraphs in black if they are direct child of <body> element. Other paragraphs put inside other elements like <div> or <td> etc. would not have any effect of this rule.

The Attribute Selectors

You can also apply styles to HTML elements with particular attributes. The style rule below will match all input elements that has a type attribute with a value of text:

```
input[type="text"]{  
  color: #000000;  
}
```

The advantage to this method is that the <input type="submit" /> element is unaffected, and the color is applied only to the desired text fields.

There are following rules applied to attribute selector.~

- ◆ p[lang] - Selects all paragraph elements with a lang attribute.
- ◆ p[lang="fr"] - Selects all paragraph elements whose lang attribute has a value of exactly "fr".
- ◆ p[lang = "fr"] - Selects all paragraph elements whose lang attribute contains the word "fr".
- ◆ p[lang="en"] - Selects all paragraph elements whose lang attribute contains values that are exactly
- ◆ "en", or begin with "en".

Multiple Style Rules

You may need to define multiple style rules for a single element. You can define these rules to combine multiple properties and corresponding values into a single block as defined in the following example:

```
h1 {  
color: #36C;  
font-weight: normal;  
letter-spacing: .4em;  
margin-bottom: 1em;  
text-transform: lowercase;  
}
```

Here all the properties and value pairs are separated by a semi colon (;). You can keep them in a single line or multiple lines. For better readability we keep them into separate lines.

Grouping Selectors

You can apply a style to many selectors if you like. Just separate the selectors with a comma as given in the following example:

```
h1, h2, h3 {  
color: #36C;  
font-weight: normal;  
letter-spacing: .4em;  
margin-bottom: 1em;  
text-transform: lowercase;  
}
```

This define style rule will be applicable to h1, h2 and h3 element as well. The order of the list is irrelevant. All the elements in the selector will have the corresponding declarations applied to them.

3.2 CSS Rules Overriding

We have discussed four ways to include style sheet rules in a an HTML document. Here is the rule to override any Style Sheet Rule.

- ◆ Any inline style sheet takes highest priority. So it will override any rule defined in `<style>...</style>` tags or rules defined in any external style sheet file.

- ◆ Any rule defined in <style>...</style>tags will override rules defined in any external style sheet file.
- ◆ Any rule defined in external style sheet file takes lowest priority and rules defined in this file will be applied only when above two rules are not applicable.

CSS Comments

Many times you may need to put additional comments in your style sheet blocks. So it is very easy to comment any part in style sheet. You simply put your comments inside `/*.....this is a comment in style sheet.....*/`.

You can use `/* ....*/` to comment multi-line blocks in similar way you do in C and C++ programming languages.

Example

```
/* This is an external style sheet file */  
h1, h2, h3 {  
  color: #36C;  
  font-weight: normal;  
  letter-spacing: .4em;  
  margin-bottom: 1em;  
  text-transform: lowercase;  
}  
/* end of style rules. */
```

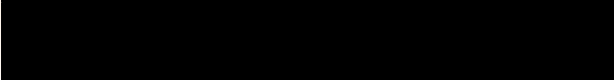
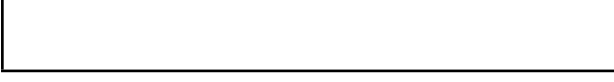
3.3 CSS Colors - Hex Codes

A hexadecimal is a 6 digit representation of a color. The first two digits(RR) represent a red value, the next two are represent a green value(GG), and the last are represent the blue value(BB).

A hexadecimal value can be taken from any graphics software like Adobe Photoshop, Jasc Paintshop Pro or even using Advanced Paint Brush.

Each hexadecimal code will be preceded by a pound or hash sign #. Following are the

examples to use Hexadecimal notation.

Color	Color HEX
	#000000
	#FF0000
	#00FF00
	#0000FF
	#FFFF00
	#00FFFF
	#FF00FF
	#C0C0C0
	#FFFFFF

3.4 CSS Colors - Short Hex Codes

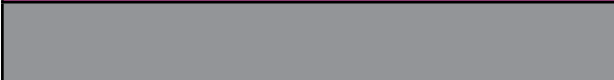
This is a shorter form of the six-digit notation. In this format, each digit is replicated to arrive at an equivalent six-digit value; For example: #6A7 becomes #66AA77.

A hexadecimal value can be taken from any graphics software like Adobe Photoshop, Jasc Paintshop Pro or even using Advanced Paint Brush. Each hexadecimal code will be preceded by a pound or hash sign #.

CSS Colors - RGB Values

This color value is specified using the `rgb( )` property. This property takes three values, one each for red, green, and blue. The value can be an integer between 0 and 255 or a percentage.

Note: All the browsers do not support `rgb()` property of color so it is recommended not to use it. Following is the example to show few colors using RGB values.

Color	Color RGB
	rgb(0,0,0)
	rgb (255,0,0)
	rgb (0,255,0)
	rgb (0,0,255)
	rgb (255,255,0)
	rgb (0,255,255)
	rgb (255,0,255)
	rgb (192,192,192)
	rgb (255,255,255)

3.4 Set the background image

Following is the example which demonstrates how to set the background image for an element.

```
<table style="background-image:url(/images/pattern1.gif);">
<tr><td>
This table has background image set.
</td></tr>
</table>
```

Set the background image position

Following is the example which demonstrates how to set the background image position 100 pixels away from the left side.

```
<table style="background-image:url(/images/pattern1.gif);  
background-position:100px;">  
<tr><td>  
Background image positioned 100 pixels away from the left.  
</td></tr>  
</table>
```

Set the background attachment

Background attachment determines whether a background image is fixed or scrolls with the rest of the page.

Following is the example which demonstrates how to set the fixed background image.

```
<p style="background-image:url(/images/pattern1.gif);  
background-attachment:scroll;">  
This paragraph has scrolling background image.  
</p>
```

3.5 Setting Fonts using CSS

This will teach you how to set fonts of a content available in an HTML element. You can set following font properties of an element:

- ◆ The font-family property is used to change the face of a font.
- ◆ The font-style property is used to make a font italic or oblique.
- ◆ The font-variant property is used to create a small-caps effect.
- ◆ The font-weight property is used to increase or decrease how bold or light a font appears.
- ◆ The font-size property is used to increase or decrease the size of a font.
- ◆ The font property is used as shorthand to specify a number of other font properties.

Set the font family

Following is the example which demonstrates how to set the font family of an element. Possible value could be any font family name.

```
<p style="font-family:georgia,garamond,serif;">  
This text is rendered in either georgia, garamond, or the default  
serif font depending on which font you have at your system.  
</p>
```

Set the font style

Following is the example which demonstrates how to set the font style of an element. Possible values are normal, italic and oblique.

```
<p style="font-style:italic;">  
This text will be rendered in italic style  
</p>
```

Set the font variant

Following is the example which demonstrates how to set the font variant of an element. Possible values are normal and small-caps.

```
<p style="font-variant:small-caps;">  
This text will be rendered as small caps  
</p>
```

Set the font size

Following is the example which demonstrates how to set the font size of an element. The font-size property is used to control the size of fonts. Possible values could be xx-small, x-small, small, medium, large, x-large, xx-large, smaller, larger, size in pixels or in %

```
<p style="font-size:20px;">  
This font size is 20 pixels  
</p>  
<p style="font-size:small;">  
This font size is small  
</p>  
<p style="font-size:large;">  
This font size is large  
</p>
```

This will produce following result:

This font size is 20 pixels

This font size is small

This font size is large

3.6 Manipulating Text using CSS

This tutorial will teach you how to manipulate text using CSS properties. You can set following text properties of an element:

- ◆ The color property is used to set the color of a text.
- ◆ The direction property is used to set the text direction.
- ◆ The letter-spacing property is used to add or subtract space between the letters that make up a word.
- ◆ The word-spacing property is used to add or subtract space between the words of a sentence.
- ◆ The text-align property is used to align the text of a document.
- ◆ The text-decoration property is used to underline, overline, and strikethrough text.
- ◆ The text-transform property is used to capitalize text or convert text to uppercase or lowercase letters.
- ◆ The white-space property is used to control the flow and formatting of text.
- ◆ the text-shadow property is used to set the text shadow around a text.

Set the text color

Following is the example which demonstrates how to set the text color. Possible value could be any color name in any valid format.

```
<p style="color:red;">
```

This text will be written in red.

```
</p>
```

Set the text direction

Following is the example which demonstrates how to set the direction of a text. Possible values are ltr or rtl.

```
<p style="direction:rtl;">  
This text will be rendered from right to left </p>
```

Set the space between characters

Following is the example which demonstrates how to set the space between characters. Possible values are normal or a number specifying space.

```
<p style="letter-spacing:5px;">  
This text is having space between letters.  
</p>
```

Set the text alignment

Following is the example which demonstrates how to align a text. Possible values are left, right, center, justify.

```
<p style="text-align:right;">  
This will be right aligned.  
</p>  
<p style="text-align:center;">  
This will be center aligned.  
</p>  
<p style="text-align:left;">  
This will be left aligned.  
</p>
```

3.7 CSS – Images

Images are very important part of any Web Page. Though it is not recommended to include lot of images but it is still important to use good images wherever it is required. CSS plays a good role to control image display. You can set following image properties using CSS.

- ◆ The border property is used to set the width of an image border.
- ◆ The height property is used to set the height of an image.
- ◆ The width property is used to set the width of an image.
- ◆ The -moz-opacity property is used to set the opacity of an image.

The image border Property

The border property of an image is used to set the width of an image border. This property can have a value in length or in %. A width of zero pixels means no border. Here is the example:

```
  
<br />  

```

This will produce following result:



Fig. 2.1

Section-3: Exercise Question

- Q. 1: Why are universal and type selectors used?
- Q. 2: What do we do when we have to apply style rule to a particular element only when it lies inside a particular element?
- Q. 3: Explain the id selector.
- Q. 4: Can we define multiple style rules?
- Q. 5: What are the rules to over ride any style sheet?
- Q. 6: How are the color codes represented in a style sheet?
- Q. 7: How do we select a background attachment?

Q. 8: Can we set the text direction? If yes then how?

Q. 9: Is rgb() property supported by all browsers?

Q. 10: Fill in the blanks:

- (a) The property is used to set the text shadow around a text.
- (b) The text-indent property is used to indent the text of a paragraph
- (c) The -moz-opacity property is used to set the of an image.

Section-4: Publishing Your website or webpages

The Final phase of creating a web site is publishing your pages. To place your pages on the web you need a web server. Commonly called hosts, web servers allow you to transfer and store files including HTML documents, images and multimedia files.

Unless you have your own web server, you need to find a server to host your pages. Some examples of webhost are:

<http://www.godaddy.com/>

<http://www.1and1.com>

<http://www.virtualempre.com>

But of course there are many more, Make sure you choose a webhost which supports FTP, which makes it much easier to publish your pages from web builder.

Page to publish

In this section of the publish window you can select which pages to publish.

Entire website

publishes all pages of the website (except the pages which have enabled 'Don't publish this page' in page properties)

Select Page Only

Publish the select page, click 'select' to specify the page to be published. Note that you can also select a folder.

Select page and sub pages

Publish the select page and all of its subpages (if any).

The 'FILES' selection specifies which files will be published.

Publish all files

All files of the selected page(s) will be published

Publish HTML files only

Select this option if you do not want to upload images or other components along with the HTML file.

Publish changed files only

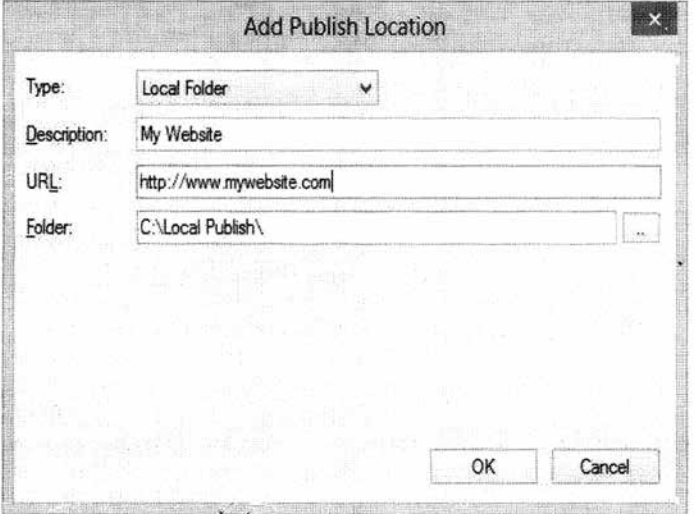
Select this option if you want to publish the files that were changed since the last time you've published your web site.

STEP:-1



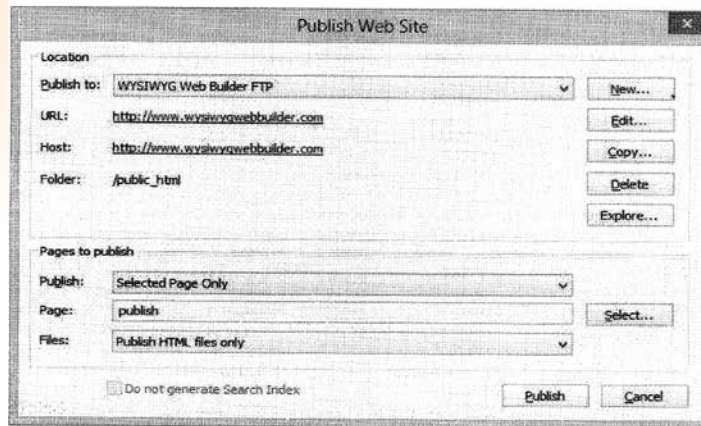
The screenshot shows the 'Add Publish Location' dialog box. The 'Type' is set to 'FTP Server'. The 'Description' is 'My Website'. The 'URL' is 'http://www.mywebsite.com'. The 'Host' is 'ftp.mywebsite.com' and the 'Port' is '21'. The 'Username' is 'loginname' and the 'Password' is masked with dots. The 'Remote folder' is '/public_html'. The checkbox 'Use passive mode for transfers' is checked. At the bottom are buttons for 'Test...', 'Explore..', 'OK', and 'Cancel'.

STEP:-2

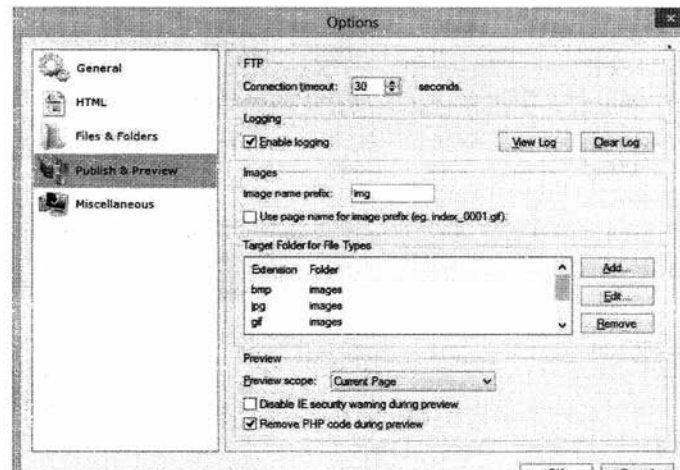


The screenshot shows the 'Add Publish Location' dialog box. The 'Type' is set to 'Local Folder'. The 'Description' is 'My Website'. The 'URL' is 'http://www.mywebsite.com'. The 'Folder' is 'C:\Local Publish\'. At the bottom are buttons for 'OK' and 'Cancel'.

STEP:-3



STEP:-4



What to do in case of publishing errors?

- ◆ Make sure your ftp address (host), username and password is valid.
- ◆ Use 'Test connection' to verify the specified data.
- ◆ Make sure you the remote folder is correct! Consult the documentation of your host for this value!
- ◆ Use of 'Explore' button to browser to the remote folder.
- ◆ If the pages don't show up on your website you've probably specified the wrong remote folder.
- ◆ Does your firewall block the FTP connection of WYSIWYG Web Builder?
- ◆ Enable 'Use passive mode for transfers'

- ◆ If the transfer takes a long time then you've probably used very large images. More details.
- ◆ Increase the connection timeout to 120 seconds or longer (Menu->Tools->Options->publish)
- ◆ If your website doesn't look to be updated, select 'Refresh' in your browser multiple times!
- ◆ Enable the publish log in Menu->Tools->Options->Publish and make note of the error message.
- ◆ For a list of all possible error codes and their descriptions go here : <http://support.microsoft.com/default.aspx?scid=kb;EN-US;193625>
- ◆ Maybe there is a temporary problem with your host? Check if you can connect using a FTP client.
- ◆ 99.99% of all publishing errors are the result of an invalid configuration, please verify the specified Data!

Section-4: Exercise Questions

- Q. 1: What do you understand by website and webpage publishing?
- Q. 2: Explain webpage, website and webserver?
- Q. 3: In how many ways you can publish your web pages?
- Q. 4: How many types of files you can publish on webserver?
- Q. 5: State five errors which commonly occur during publishing of any website?

Further Readings

Book-1 Kogent Learning Solutions, "HTML BLACK BOOK," 4th Edition, Dreamtech Press, 2012

Book-2 Elisabeth R, Eric F, "Head first HTML and CSS," 2nd Edition, O'REILLY, 2013

Website 1: www.w3schools.com

Website 2: www.thenewboston.com

Website 3: www.godaddy.com

Website 4: www.freesever.com

3

WEB SCRIPTING JAVASCRIPT

Learning Objectives

Section-1: Creating interactive Web Pages with scripts

Section-2: Variables and Operators

Section-3: Decision making using if and switch

Section-4: Iterations - loops

Section-5: Window Object

Section-6: Popup Boxes — alert, confirm, prompt

Section-7: Arrays

Section-8: Functions

Section-1: Creating interactive Web Pages with Javascripts

1.1 JavaScript Basics

This section deals with the basics of JavaScript covering syntax, semantics, enabling JavaScript browser, creating simple web pages, writing your first JavaScript and displaying in a web browser.

Objectives

After studying this section the students will be able to:

- ◆ Use a text editor such as Notepad to publish an HTML document
- ◆ recall the HTML concepts
- ◆ Learn the history and uses of JavaScript

- ◆ Understand the JavaScript syntax and structures.
- ◆ Learn, how JavaScript programs are used in a web pages including the use of event-handlers and the Document Object Model (DOM)
- ◆ Insert JavaScript into a web page using various approaches including inline code, internal scripts and external JavaScript files
- ◆ Write and use simple JavaScript functions and event handlers that reflect common applications of JavaScript
- ◆ Acquire confidence in the use of JavaScript in a web page with the help of given case study

Difference between Arrays and Objects:

- ◆ In JavaScript, arrays use numbered indexes whereas objects use named indexes.
- ◆ Arrays are a special kind of objects with numbered indexes.
- ◆ JavaScript does not support associative arrays.
- ◆ You should use objects when you want the element names to be string (text).
- ◆ You should use arrays when you want the element names to be number.

Prerequisites

- ◆ The user need to use any text editor like Notepad and an Internet browser such as Internet Explorer/Firefox/Google Chrome or Opera
- ◆ Hardware: Any Computer System
- ◆ Software: Windows OS

Q. What is browser? What are the major types of browser available?

Ans. A browser is an application program with a graphical user interface for displaying HTML files, used to navigate the World Wide Web which provides a way to look at and interact with all the information on the World Wide Web. The two most popular browsers are Microsoft Internet Explorer and Firefox. Other major browsers include Google Chrome, Apple Safari and Opera.

Q. How to make browser a JavaScript enabled browser? How to enable JavaScript in a web browser?

Ans. JavaScript is integrated into the browsing environment, which can get information about the browser and HTML pages, and can modify this information to, thus changing how things are presented on your screen.

Enabling JavaScript in Browsers

All the modern browsers are available with built-in support for JavaScript. However, sometimes you may need to enable or disable this support manually. Here we explain the procedure of enabling and disabling the JavaScript support in different browser such as Internet Explorer, Firefox, Chrome, and Opera.

Setting JavaScript in Internet Explorer

Following are the simple steps to turn on or turn off JavaScript in your Internet Explorer:

1. Select **Tools** → **Internet Options** from the menu.
2. Select **Security** tab from the dialog box.
3. Click the **Custom Level** button.
4. Scroll down till you find **Scripting option**.
5. Select *Enable* radio button under **Active scripting**.
6. Finally click OK

To disable JavaScript support in your Internet Explorer, you need to select **Disable** radio button under **Active scripting**.

Setting JavaScript in Firefox

Here are the steps to turn on or turn off JavaScript in Firefox :

1. Open a new tab → type **about: config** in the address bar.
2. Then you will find the warning dialog. Select **I'll be careful, I promise!**
3. Then you will find the list of **configure options** in the browser.
4. In the search bar, type **JavaScript enabled**.
5. Now, you will find an option to enable or disable JavaScript by right-clicking on the value of that option → **select toggle**.

If JavaScript enabled is true; it converts to false by clicking **toggle**. If JavaScript is disabled; it gets enabled by clicking toggle.

Setting JavaScript in Chrome

Here are the steps to turn on or off JavaScript in Chrome:

1. Click the Chrome menu at the top right corner of your browser.
2. Select **Settings**.
3. Click **Show advanced settings** at the end of the page.

4. Under the **Privacy** section, click the Content settings button.
5. In the “JavaScript” section, select “Do not allow any site to run JavaScript” or “Allow all sites to run JavaScript (recommended)”.

Setting JavaScript in Opera

Here are the steps to turn on or turn off JavaScript in Opera:

1. Select **Tools** → **Preferences** from the menu.
2. Select **Advanced** option from the dialog box.
3. Select **Content** from the listed items.
4. Select **Enable JavaScript** checkbox.
5. Finally click OK and come out.

To disable JavaScript support in Opera, do not select the **Enable JavaScript** checkbox.

What is JavaScript?

JavaScript is an interpreted computer programming language.

- ◆ It was originally implemented as a part of web browser so that client-side scripts could interact with the user, control the browser, communicate asynchronously, and alter the contents of the document.
- ◆ JavaScript is not useful as a standalone language, but is designed for easy embedding in other products and applications, such as web browsers.
- ◆ JavaScript contains a core set of objects such as Array, Date, Math, and a core set of language elements like operators, control structures, and statements.

History

JavaScript was developed in 1995 by Brendan Eich, at Netscape, and first released with Netscape 2 early in 1996. It was initially called LiveScript, but was renamed JavaScript to capitalize the popularity of Sun Microsystems’s Java language. The basic purpose of JavaScript language was to enable web authors to design interactive sites. JavaScript can interact with HTML source code, enabling web authors to modify their sites with dynamic contents. JavaScript is supported by a number of software companies and is an open language that anyone can use without purchasing a license. JavaScript language is interpreted and executed by the browser.

Standards

In November 1996, Netscape announced JavaScript to European Computer Manufacturers Association (ECMA) for consideration as an industry standard and subsequent work resulted in the standardized version named ECMAScript. ECMA is an international standard organization for information and communication system.

- ◆ In June 1997, ECMA International published the first edition of ECMA-262 specification.
- ◆ A year later, in June 1998, some modifications were made to adapt it to the ISO/IEC-16262 standard, and the second edition was released.
- ◆ The third edition of ECMA-262 is the version most browsers are currently using
- ◆ Fourth edition of ECMAScript standard was not released and does not exist in practice
- ◆ Fifth edition of the ECMAScript standard was released in December 2009
- ◆ The current edition of ECMAScript standard is 5.1 and it was released in June 2011.

This way, JavaScript has become one of the most popular programming language on the web. JavaScript was formalized in the ECMAScript language standard and primarily used as a part of web browsers (client-side JavaScript). This enables programmer to access the computational objects within a host environment. JavaScript gained a widespread success instantly/quickly as a client-side scripting language for web pages. Microsoft introduced JavaScript support in its own web browser, Internet Explorer, in version 3.0, released in August 1996.

Features

JavaScript can handle simple as well as complex tasks that makes it a robust language. It is also a good language for web designers & learners who wish to learn computer programming as it supports both object oriented and function oriented concepts and to use it, you just need a browser and a text editor. Some specific features of JavaScript are discussed below.

- ◆ Browser support: All browsers accept JavaScript as a scripting language and provide integrated support for it. For example, to access flash content, you need to install flash and plug-in in your browser. But to use JavaScript, you don't have to use any plug-in at all.

- ◆ JavaScript can be used on client side as well as on server side: JavaScript has access to Document Object Model (DOM) of the browser. You can change the structure of web pages at a runtime. Thus, JavaScript can be used to add different effects to web pages. On the other hand, JavaScript can be used on the server side as well.
- ◆ Functional programming language: In JavaScript, a function can be assigned to variables just like any other data types. A function can accept another function as a parameter and can also return a function. This provides you the ability to code in functional programming style.
- ◆ Support for objects: JavaScript is an object oriented language. However, the way JavaScript handles objects and inheritance is a bit different from the conventional object oriented programming languages like C++/ Java. JavaScript supports most of the object oriented concepts while being simple to learn and use.
- ◆ **Run-time environment:** JavaScript typically relies through upon a run-time environment (e.g., in a web browser) to provide objects and methods by which scripts can interact with the outside world. In fact, it relies upon the environment to provide the ability to include/import scripts (e.g. HTML <script> elements).
- ◆ **Vendor-specific extensions:** JavaScript is officially managed by Mozilla Foundation, and new language features are added periodically. However, only some JavaScript engines support these new features.
- ◆ **Object based Features Supported by JavaScript:** JavaScript supports various features of object based languages and JavaScript is sometimes referred as an object-based programming language.

Advantages

- ◆ JavaScript gives HTML designers a programming tool with a very simple syntax. So that the designers can put small “snippets” of code into their HTML pages
- ◆ JavaScript can put dynamic text into an HTML page.
- ◆ JavaScript can react to events. For eg. a JavaScript can be set to execute when something happens, like when a page has finished loading or when a user clicks on an HTML element.
- ◆ JavaScript can read and change the contents of an HTML element.
- ◆ JavaScript can be used to validate data of the form before it is submitted to a server. It saves the server from extra processing.

- ◆ JavaScript can be used to create cookies. For example, a JavaScript can be used to store and retrieve information from the visitor's computer.
- ◆ JavaScript is widely used for developing modern web applications with the capabilities of desktop applications.
- ◆ JavaScript supports the best practices for code organization, modularisation, and reuse.
- ◆ Using JavaScript, you can divide your application into different layers of responsibility.
- ◆ JavaScript build efficient, self-describing hypermedia Application Programming Interface (APIs) with Node.js.
- ◆ JavaScript can be used to test, integrate, and deploy software updates in rapid cycles.
- ◆ JavaScript control resource access with user authentication and authorization.

Applications

- ◆ Developing multimedia applications: The users can use JavaScript to add multimedia elements. With JavaScript you can show, hide, change, resize images and create images rollovers. You can create scrolling text across the status bar, thus making multimedia applications more interactive.
- ◆ Create pages dynamically: Based on the user's choice, JavaScript can generate pages that are customized by the user.
- ◆ Interact with the user: JavaScript can do some processing of forms and can validate user input when the user submits the form.
- ◆ JavaScript objects are similar to dictionaries: In JavaScript, objects are just a collection of name-value pairs. JavaScript objects are considered as a dictionary with string keys. The users can get and set the properties of an object using either the familiar "." (dot) operator, or the "()" operator, which is typically used when dealing with a dictionary.
- ◆ JavaScript can be extended for different purposes by supplementing it with additional objects.
- ◆ Client-side JavaScript extends the core language by supplying objects to control a browser (Navigator or another web browser) and its Document Object Model (DOM). For example, client-side extensions allow an application to place elements on an HTML form and respond to the user events such as mouse clicks, form input, and page navigation.

- ◆ Server-side JavaScript extends the core language by supplying objects relevant to running JavaScript on a server. For example, server-side extensions allow an application to communicate with a relational database, which provide continuity of information from one invocation to another of the application, or perform file manipulations on a server.

Note: JavaScript and Java are completely different programming languages, both in concept and design. JavaScript was invented by Brendan Eich in 1995, and became an ECMA standard in 1997. Java, developed by Sun Microsystems is powerful programming language in the category of object oriented languages.

Why to use JavaScript?

- ◆ JavaScript is a simple scripting language invented specifically to use in web browsers to make websites more dynamic.
- ◆ HTML is capable of outputting more-or-less static pages. Once you load them up, the view doesn't change much until you click a link to go to a new page. Whereas, adding JavaScript to your code allows you to change how the document looks, from changing text, to changing colours, to changing the options available in a drop-down list and much more.
- ◆ JavaScript is a client-side language, which means all the action occurs on the client's (user's) side of things.
- ◆ JavaScript operations are usually performed instantaneously. In fact, JavaScript is often used to perform operations that would otherwise restrain the server, like form input validation. This load distribution of work to the client-side services speed up the process significantly.

1.2 Javascript Syntax

JavaScript syntax is the set of rules that define a structured JavaScript. Here are some tips to remember when writing your first JavaScript program.

Case Sensitivity

JavaScript is case sensitive i.e., upper case letter and lower case letter has different meaning. For example, the word "alert" has a lower case "a". So, if we type the word with an uppercase "A" then JavaScript will show an error and the alert box will not be displayed.

Whitespace & Semi Colon

JavaScript ignores spaces, tabs, and newlines that appear in JavaScript programs. You can use spaces, tabs, and newlines freely in your program and you are free to format and indent the programs in a neat and consistent way that makes the code easy to read and understand.

The following code could be written without semicolons.

```
<script language="javascript" type="text/javascript">
<!--
var1 = 10
var2 = 20
//-->
</script>
```

But when formatted in a single line as follows, you must use semicolons –

```
<script language="javascript" type="text/javascript">
<!--
var1 = 10; var2 = 20;
//-->
</script>
```

Example Explained

Whitespaces (usually spaces & tabs) are used for the better understanding & readability of the code to the user. Therefore, it is a good programming practice to leave spaces and write code legibly. A semicolon is a line or statement terminator in JavaScript. It is optional and you may skip to use semicolons in your program. JavaScript will not show an error unless two statements are merged into a single line.

Note: It is a good programming practice to use semicolons.

Comments

Comments, written by preceded with two forward slashes are used to improve the readability of program and will be ignored by the interpreter. There are two kind of comments in JavaScript: *Single line* and *Multi line* comments

1. Single line comment begins with two forward slashes // This is a single line comment
2. Multiline comments begin with /* and ends with */

e.g., /* This is a multi line comment

and may continue across multiple lines */

Comments are ignored by the interpreter and are not executed. Comments play an important role in helping the programmer to understand complex logic while writing JavaScript code.

Line Break

Line breaks are ignored except when it is used within a statement

JavaScript Code

- ◆ JavaScript code is typically embedded with in the HTML. It is interpreted and run by the client's browser.
- ◆ The <SCRIPT> tag alerts a browser that JavaScript code is embedded with in the HTML. It is described below:

<SCRIPT language = "JavaScript">	// Beginning of the SCRIPT tag
Statement-1; Statement-2;	// JavaScript statements, functions, variables etc.
</SCRIPT>	// Ending of the SCRIPT tag

Note: // the comments in the above example, preceded with two forward slashes, are used to improve the readability of program and will be ignored by the interpreter.

1.3 Writing First Java Program

JavaScript is written in the same way as HTML, in a text-editor like notepad. JavaScript implementation is quite similar to CSS (Cascading Style Sheets). The user can link it to the outside files (with the file extension .js), or write blocks of code into their HTML documents with the <script> tag.

Here, we consider a simple example with an **embedded script**. This will simply display a text *Hello World* on the screen.

<script type="text/javascript">

<!--

```
document.write("<i>Hello World!</i>");
```

//My First JavaScript Program to print Hello World on the screen

</script>

When you place this in your code the text Hello World will appear on the screen i.e.,

Hello World!

The 'script' tag encloses any script code you want to use. The 'type' attribute is used to alert the browser to the type of script it is about to deal with, and so helps it to interpret the JavaScript code.

Consider another example:

```
<html>
<head>
<title>My Javascript Page</title>
</head>
<body>
<script type="text/javascript">
alert("Welcome to my world!!!");
</script>
</body>
</html>
```

Output: Welcome to my world!!!

Example explained

How to save and run your first program in JavaScript

1. Open any editor such as notepad and write the program
2. Save the program with .html extension in a proper folder or subfolder on a drive like C:\JavaScript\myprograms.
3. Open the web browser like internet explorer or Mozilla Firefox
4. Open the file you have created and saved in step 2, and execute the program. For eg. execute C:\JavaScript\firstprogram.html

Where C is the drive, JavaScript is the folder name and firstprogram.html is the name of program which is saved in the JavaScript folder

To indicate that the given code is written in JavaScript, you need to place the code within **<script> </script>** HTML tags.

Note: JavaScript does not have any built-in print or display functions. The following method can be applied using JavaScript display.

- ◆ Writing into an alert box, using **window.alert()**.
- ◆ Writing into the HTML output using **document.write()**.
- ◆ Writing into an HTML element, using **innerHTML**.
- ◆ Writing into the browser console, using **console.log()**.

Using window.alert():

Try It Out!

Type the following text in editor and observe the output:

```
<!DOCTYPE html>
<html>
<body>
<h1>My First Web Page</h1>
<p>My first paragraph.</p>
<script>
window.alert("Hello"); // Any number or value can be defined
</script>
</body>
</html>
```

Using document.write():

Try It Out!

Type the following text in editor and observe the output:

```
<!DOCTYPE html>
<html>
<body>
<h1>My First Web Page</h1>
<p>My first paragraph.</p>
<script>
document.write(100); // Any number or value can be defined
</script>
</body>
</html>
```

Using document.write()

Try It Out!

Type the following text in editor and observe the output:

```
<!DOCTYPE html>
<html>
<body>

<h1>My First Web Page</h1>
<p>My first paragraph.</p>
<button onclick="document.write(5 + 6)">Try it</button>
// Any number or value can be defined
</body>
</html>
```

Using inner HTML

To access an HTML element, JavaScript can use the document[.]getElementById(id) method. The id attribute defines the HTML element. The innerHTML property defines the HTML content:

Try It Out!

Type the following text in editor and observe the output

```
<!DOCTYPE html>
<html>
<body>
<h1>My First Web Page</h1>
<p>My First Paragraph</p>
<p id="demo"></p>
<script>
document.getElementById("demo").innerHTML = 100;
// Any number or value can be defined
</script>
</body>
</html>
```

Using console.log()

In your browser, you can use the **console.log()** method to display data. Activate the browser console by pressing F12 key, and select "Console" in the menu.

Try It Out!

Type the following text in editor and observe the output

```
<!DOCTYPE html>
<html>
<body>
<h1>My First Web Page</h1>
<p>My first paragraph.</p>
<script>
console.log(5 + 6);
</script>
</body> </html>
```

Summary

JavaScript language has no concept of input or output. It is designed to run as a scripting language in a host environment. It is up to the host environment to provide mechanisms for communicating with the outside world. The most commonly used host environment

is the browser. JavaScript interpreters can also be found in other formats such as Adobe Acrobat, Photoshop, Yahoo!'s Widget engine, and server side environments. JavaScript programs manipulate values, and all these values belong to a type. Different JavaScript's types include Number, String, Boolean and Objects.

There is no built-in I/O functionality in JavaScript. It is the runtime environment which provides I/O functionality to the user. However, most runtime environments have a console object that can be used to print output.

Section-1: Exercise Questions

Perform the following activities in lab:

S. No.	Activities
1.	Write a program with an alert box to: a) Display the text "Good Morning Sir". The JavaScript code must be in the <head> tag. b) Display the text "I am learning JavaScript". The JavaScript code must be in an another file called "MyJavascript.js"
2.	Write a program to display your name, school and class teacher with an onload event.
3.	Write a program to display an alert box which says "Welcome to the world of computing"
4.	Write a program to display a confirmation box which says "Did you enjoy Learning JavaScript session?"
5.	Write a program to display a prompt box which says "Please give your feedback here"

Answer the following questions

- Q.1: What is an editor?
- Q.2: What do you understand by a web browser?
- Q.3: What are the major types of browser available today?
- Q.4: What is the advantage of using Javascript over HTML?
- Q.5: What are the steps needed to create and run a simple program using JavaScript?

- Q. 6: Which attribute of the <script> tag helps you include an external JavaScript file?
- Q. 7: What are the three ways in which JavaScript can be included?
- Q. 8: JavaScript is “interpreted” - what does this imply?
- Q. 9: Explain the role of case sensitivity in JavaScript.
- Q. 10: How to use multiple line comments in Javascript.
- Q. 11. Fill in the blanks
- (a) A file which ends with .js is an
 - (b) JS programs are included within & of a HTML document.
 - (c) Generally HTML is preferred for creating while CSS is preferred for and JavaScript is preferred for
 - (d) JavaScript Syntax is a set of that define a structured JavaScript.
 - (e) Comments are used to in JavaScript program.

Further Readings

1. <http://www.javascriptkit.com>
2. <https://developer.mozilla.org/en/docs/JavaScript>
3. <http://www.dynamicdrive.com>

Section-2: Variables and Operators

Variables and Operators

This section deals with creating variables, naming, types of variables and different types of operators supported by Javascript and how to use them

2.1 Variables

- ◆ Variables can be thought of as named containers in the computer’s memory. You can place data in these containers and refer to the value using the name.
- ◆ A variable consists of an identifier, a scope and a specific data type.
- ◆ **An Identifier** is the name of a variable.

- ◆ Identifier must start with a character, a dollar sign, or an underscore.
- ◆ An identifier can't use reserved keywords.

Table: List of reserved keywords in JavaScript

break	else	new	var
case	final	return	void
catch	for	switch	while
continue	function	this	with
default	if	throw	delete
in	try	do	instance of
type of	interface	null	undefined

Variable Declaration

- ◆ Variables in JavaScript can be defined using the keyword **var**.
- ◆ The equal to (=) sign is used to **assign** a value to a variable.
- ◆ Note that **var** is the keyword in JavaScript.
- ◆ Users can either, separately declare the variable and then assign values to it or straight-away declare and initialize the variables.

Examples of variables

```

var X;           //defines a variable X, and by default no value is assigned to this
                  variable
var Y = 100;     //defines a variable Y and assigns the value of 100 to it
var customer     // declare the variable using the var keyword
X = 10;          // use the = operator to assign a value
var x = 10;      //assigning a value to a variable declaring it
var y=5;         //y is a variable that holds a value of 5
var sum = x + y ; // sum is a variable that stores the sum of variables x and y

```

Try It Out!

```
<!DOCTYPE html>
<html>
<body>
<h1>JavaScript Variables Example</h1>
<p>x = 5, y = 2, calculate z = x + y, and display z:</p>
<p id="demo"></p>
<script>
var x = 5;
var y = 2;
var z = x + y;
document.getElementById("demo").innerHTML = z;
</script>
</body>
</html>
```

Output: **JavaScript Variables Example**

x = 5, y = 2, calculate z = x + y, and display z:
7

Variable Scope

The scope of a variable refers to the area of a program in which it can be accessed. Variables have two types of scopes -global scope and local scope.

The Global Scope: Consider the example while handling the rate of sugar, we can declare a variable SugarRate at global. It is fixed in the fair price shop by the government and is same throughout the state in all the fair price shops. This is an example of a variable named SugarRate being declared globally.

The Local Scope: In case of local scope the rate of sugar in different retails shops will be specific to the rating norms of each shop. This is an example of a local variable. It is specific within the scope of each shop.

2.2 Data Types

A **data type** is a classification of the type of data that a variable or an object can hold.

Data type is an important factor in all computer programming languages, including visual basic, C#, C/C++ and JavaScript. JavaScript supports different data types such as Numbers, Strings, Boolean, Arrays and Objects listed in the Table as below:

Data Type	Examples	Description
Numbers	183, 574.445, 858.3, 1.0555e3	integers, decimals and extra large/extra small numbers with scientific notation
String	“This is a valid string”	series of characters within double quotes
Boolean	True, False	True or False values. It is used for conditional testing

Note: You can use quotes inside a string as long as they are not the same as the quotes surrounding the string. For example, consider a string “Aditya’s turn to take the Ride now” is accepted as a string whereas the string “Her name is “Reena” and she will be here soon” is not a valid string.

Numbers

Numbers in JavaScript are double-precision 64-bit format. The standard numeric operators are supported, including addition, subtraction, modulus (or remainder) arithmetic and so forth.

Strings

Strings in JavaScript are sequences of characters. More precisely, they're sequences of Unicode characters, with each character represented by a 16-bit number. If we want to represent a single character, we just need to use a string of length 1.

To find the length of a string, access its length property:

```
"hello".length  returns 5           //length of given string "hello"
```

The strings are represented using objects and they have methods as well:

```
"hello, world".replace("hello", "goodbye") //by using replace() method goodbye, world
```

```
“hello”.toUpperCase() //by using toUpperCase() method HELLO
```

Operators

An Operator is a symbol that is used to perform an operation. In this section we will learn how to use JavaScript operators.

2.3 Assignment operators

Assignment operators help in assigning values to a variable. The following table shows how assignment operators work in JavaScript.

Here we consider two variables x and y assuming x=90 and y=20. Upon execution, the assignment operator will generate the following results.

Operator	Example	Same As	Result
=	x=y		x=90
+=	x+=y	x=x+y	x=110
-=	x-=y	x=x-y	x=70
* =	x*=y	x=x*y	x=1800
/=	x/=y	x=x/y	x=4.5
%=	x%=y	x=x%y	x=1

The equal to (=) sign is used to **assign** a value to a variable.

Arithmetic Operators

There are following arithmetic operators supported by JavaScript language: Assume variable A holds 10 and variable B holds 20 then:

Operator	Description	Example
+	Adds two operands	A + B will give 30
-	Subtracts second operand from the first	A - B will give -10
*	Multiply both operands	A * B will give 200
/	Divide numerator by denominator	B / A will give 2
%	Modulus Operator and remainder of after an integer division	B % A will give 0
++	Increment operator, increases integer value by one	A++ will give 11
--	Decrement operator, decreases integer value by one	A-- will give 9

Comparison Operators

There are following comparison operators supported by JavaScript language. Assume variable A holds 10 and variable B holds 20 then

Operator	Description	Example
==	Checks if the values of two operands are equal or not, if yes then the condition becomes true.	(A == B) is not true.
!=	Checks if the values of two operands are equal or not, if values are not equal then condition becomes true.	(A != B) is true.
>	Checks if the value of left operand is greater than the value of right operand, if yes then the condition becomes true.	(A > B) is not true.
<	Checks if the value of left operand is less than the value of right operand, if yes then the condition becomes true.	(A < B) is true.
>=	Checks if the value of left operand is greater than or equal to the value of right operand, if yes then the condition becomes true.	(A >= B) is not true.
<=	Checks if the value of left operand is less than or equal to the value of right operand, if yes then the condition becomes true.	(A <= B) is true.

Logical Operators

There are following logical operators supported by JavaScript language. Assume variable A holds 10 and variable B holds 20 then:

Operator	Description	Example
&&	Called Logical AND operator. If both the operands are non-zero then condition becomes true.	(A && B) is true.
	Called Logical OR Operator. If any of the two operands are non-zero then the condition becomes true.	(A B) is true.
!	Called Logical NOT Operator. Use to reverse the logical state of its operand. If a condition is true then Logical NOT operator will make it false.	!(A && B) is false.

Bitwise Operators

There are following bitwise operators supported by JavaScript language. Assume variable A holds 2 and variable B holds 3 then:

Operator	Description	Example
&	Called Bitwise AND operator. It performs Boolean AND operation on each bit of its integer arguments.	(A & B) is 2.
	Called Bitwise OR Operator. It performs Boolean OR operation on each bit of its integer arguments.	(A B) is 3.
^	Called Bitwise XOR Operator. It performs Boolean exclusive OR operation on each bit of its integer arguments. Exclusive OR means that either operand one is true or operand two is true but not both.	(A ^ B) is 1.
~	Called Bitwise NOT Operator. It is a unary operator and operates by reversing all bits in the operand.	(~B) is -4.
<<	Called Bitwise Shift Left Operator. It shift all the bits in its first operand to the left by the number of places specified in the second operand. New bits are filled with zeros. Shifting a value left by one position is equivalent to multiplying by 2, shifting two positions is equivalent to multiplying by 4 so on.	(A << 1) is 4.
>>	Called Bitwise Shift Right with Sign Operator. It shift all the bits in its first operand to the right by the number of places specified in the second operand. The bits filled in on the left depend on the sign bit of the original operand, in order to preserve the sign of the result. If the first operand is positive, the result has zeros placed in the high bits; if the first operand is negative, the result has ones placed in the high bits. Shifting a value right one place is equivalent to dividing by 2 (discarding the remainder), shifting right two places is equivalent to integer division by 4, and so on.	(A >> 1) is 1.

>>>	Called Bitwise Shift Right with Zero Operator. This operator is just like the >> operator, except that the bits shifted in on the left are always zero,	(A >>> 1) is 1.
-----	---	-----------------

Assignment Operators

There are following assignment operators supported by JavaScript language:

Operator	Description	Example
=	Simple assignment operator, Assign values of right side operands to the left side operand	C = A + B will assign value of A + B to C
+=	Add AND assignment operator, It adds the right operand to the left operand and assign the result to the left operand	C += A is equivalent to C = C + A
-=	Subtract AND assignment operator, It subtracts right operand from the left operand and assign the result to the left operand	C -= A is equivalent to C = C - A
*=	Multiply AND assignment operator, It multiplies the right operand with the left operand and assign the result to the left operand	C *= A is equivalent to C = C * A
/=	Divide AND assignment operator, It divides the left operand with the right operand and assign the result to the left operand	C /= A is equivalent to C = C / A
%=	Modulus AND assignment operator, It takes modulus using two operands and assign the result to the left operand	C %= A is equivalent to C = C % A

Miscellaneous Operator

◆ Conditional Operator (? :)

This operator first evaluates an expression for a true or false value and then execute one of the two given statements depending upon the result of the evaluation.

Operator	Description	Example
? :	Conditional Expression	If Condition is true? Then value X : Otherwise value Y

◆ **typeof Operator**

The *typeof* is a unary operator that is placed before a single operand, which can be of any type. Its value is a string indicating the data type of the operand.

The *typeof* operator evaluates to “number”, “string”, or “boolean” if its operand is a number, string, or boolean value and returns true or false based on the evaluation.

Here is the list of return values for the *typeof* Operator:

Type	String returned by typeof
Number	“number”
String	“string”
Boolean	“boolean”
Object	“object”
Function	“function”
Undefined	“undefined”
Null	“object”

Section-2: Exercise Questions

Perform the following activities in lab.

S. No.	Activities
1.	Write a program to show the use of arithmetic operators in JavaScript.
2.	Write a program in JavaScript to check if the value of two operands is equal or not.
3.	Write a program in JavaScript to implement Bitwise operators.
4.	Write a program in JavaScript to implement AND, OR, NOT operators.
5.	Write a JavaScript program to: ◆ Add two numbers and display the sum in an alert box. ◆ Subtract two numbers and display the difference

Answer the following questions:

Q.1: What is a variable?

Q.2: Differentiate between local and global scope of a variable.

Q.3: What is an operator?

- Q.4: What are the major types of operators supported by Javascript?
- Q.5: What are undeclared and undefined variables?
- Q.6: What is a string?
- Q.7: What would be the result of 1+2+"3" in JavaScript?
- Q.8: What is the difference in the outputs that you get when you use the following operators =; == ; and ===.
- Q.9: Explain what you understand by Modulo operator.
- Q.10: Specify four types of operators that can be used in JavaScript.
- Q.11: Explain what a logical NOT operator does using some examples.
- Q.12: What is conditional operator?
- Q.13: Fill in the blanks
- (a) If $x = 24$ and $y = 3$ then the output of $x * x / y$ is
 - (b) if $x = 25$ and $y = 3$ then the output of $x += y$ is
 - (c) , and are all logical operators used in Javascript
 - (d) If $x = 14$ and $y = 3$ then the output of $x \bmod y$ is
 - (e) if $x = 15$ and $y = 3$ then the output of $x \% y$ is
 - (f) Plus sign is used to numbers and strings.
 - (g) When I want to display an alert which says 'Welcome' and then the name of the student, I will have to use the operator.

Further Readings

1. <http://www.javascriptkit.com>
2. <https://developer.mozilla.org/en/docs/JavaScript>
3. <http://www.dynamicdrive.com>

Section-3: Decision making using if and switch

3.1 Control Structures in JavaScript

JavaScript has a similar set of control structures like the other languages such as C, C++ and Java. Conditional statements are supported by if and else described below:

Conditional Statements

In JavaScript we have the following conditional statements as depicted in fig. 3.1:

- ◆ **if** (statement) specify a block of code to be executed, if the specified condition is true
- ◆ **else** (statement) specify a block of code to be executed, if the same condition is false
- ◆ **else if** (statement) specify a new condition to test, if the first condition is false
- ◆ **switch** (statement) specify many alternative blocks of code to be executed

Syntax: **if** statement

```
if (condition) {
```

Note: if is used in lowercase, Uppercase letters like IF or iF or If will generate a JavaScript error

block of code to be executed if the condition is true

```
}
```

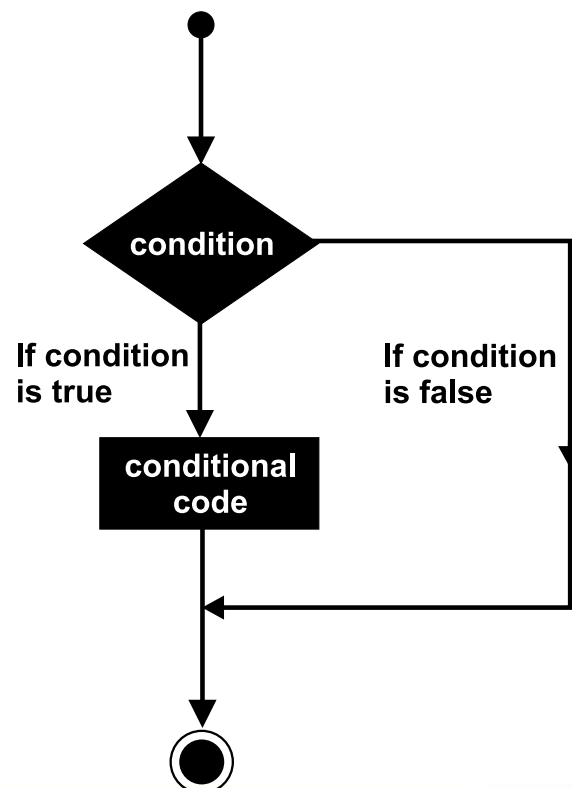


Fig. 3.1

Example: Make a “Happy day” greeting if the hour is less than 18:00:

```
<!DOCTYPE html>
<html>
<body>
<p>Display “Good day!” if the hour is less than 18:00:</p>
<p id=“demo”>Good Evening!</p>
<script>
if (new Date().getHours() < 18) {
    document.getElementById(“demo”).innerHTML = “Happy day!”;
}
</script>
</body>
</html>
```

Output: Display “Happy day!” if the hour is less than 18:00:
Happy day!

The **else** statement can be used to specify a block of code to be executed if the same condition is false.

Syntax: **else** statement

```
if (condition) {
    block of code to be executed if the condition is true
} else
{
    block of code to be executed if the condition is false
}
```

Example:

If the hour is less than 18, create a “Good day” greeting, otherwise “Good evening”:

```

<!DOCTYPE html>
<html>
<body>
<p>Click the button to display a time-based greeting:</p>
<button onclick="myFunction()">Try it</button>
<p id="demo"></p>
<script>
function myFunction() {
    var hour = new Date().getHours();
    var greeting;
    if (hour < 18) {
        greeting = "Good day";
    } else {
        greeting = "Good evening";
    }
    document.getElementById("demo").innerHTML = greeting;
}
</script>
</body>
</html>

```

Output: The above script will display “Good day or Good evening” depending upon the current time of the day

The ‘else if’ Statement is used to specify a new condition if the first condition is false.

Syntax:

```

if (condition1)
{
    block of code to be executed if condition1 is true
} else if (condition2) {
    block of code to be executed if the condition1 is false and condition2 is true
} else {
    block of code to be executed if the condition1 is false and condition2 is false
}

```

Example:

If time is less than 10:00, display a “Good morning” greeting, if not, but time is less than 20:00, display a “Good day” greeting, otherwise a “Good evening”: greeting

```
<!DOCTYPE html>
<html>
<body>
<p>Click the button to get a time-based greeting:</p>
<button onclick="myFunction()">Try it</button>
<p id="demo"></p>
<script>
function myFunction() {
    var greeting;
    var time = new Date().getHours();
    if (time < 10) {
        greeting = "Good morning";
    } else if (time < 20) {
        greeting = "Good day";
    } else {
        greeting = "Good evening";
    }
    document.getElementById("demo").innerHTML = greeting;
}
</script>
</body>
</html>
```

Output: Display “Good morning or Good day or Good evening” depending upon the current time of the day

The **switch** statement is used to select one of many blocks of the code to be executed in a program described below. The switch expression is evaluated once and the value of the expression is compared with the values of each case. If there is a match found, then the associated block of code is executed otherwise default code block is executed.

Syntax: switch statement (as depicted in figure 3.2)

```
switch (expression)
{
    case n:
        code block
        break;
    case n:
        code block
        break;
    default:
        default code block
}
```

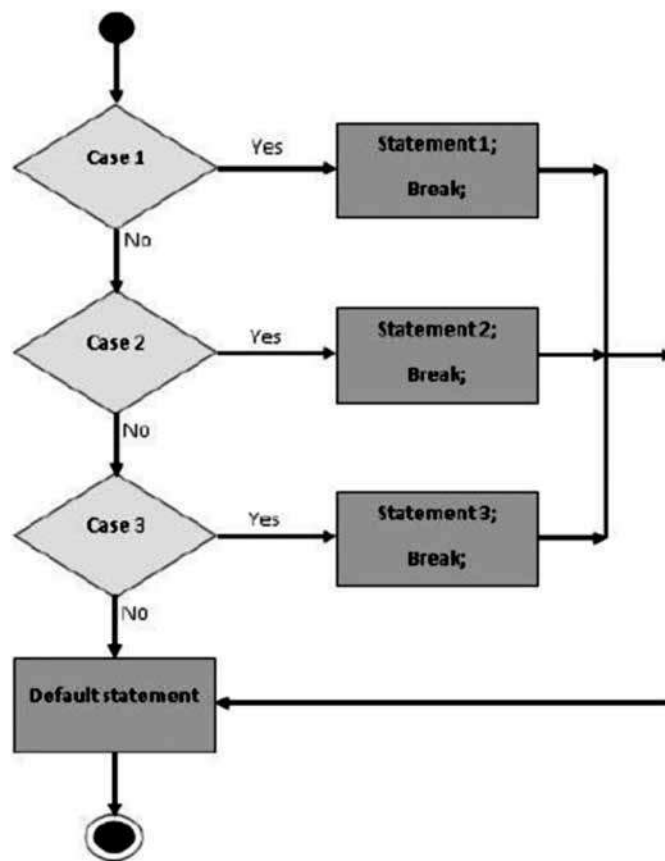


Fig. 3.2

Example

Use the weekday number to calculate weekday name with the help of `getDay()` method. The `getDay()` method returns the weekday as a number between 0 and 6. (Sunday=0, Monday=1, Tuesday=2 ..)

```
<!DOCTYPE html>
<html>
<body>
<p id="demo"></p>
<script>
var day;
switch (new Date().getDay()) {
  case 0:
    day = "Sunday";
    break;
  case 1:
    day = "Monday";
    break;
  case 2:
    day = "Tuesday";
    break;
  case 3:
    day = "Wednesday";
    break;
  case 4:
    day = "Thursday";
    break;
  case 5:
    day = "Friday";
    break;
  case 6:
    day = "Saturday";
    break;
}
document.getElementById("demo").innerHTML = "Today is " + day;
</script>
</body>
</html>
```

Output: Today is Sunday (from Sunday to Saturday each single day will be displayed depending upon current day of the week)

Section-3: Exercise Questions

Perform the following activities in Lab.

S. No.	Activities
1.	Write a program to: ◆ Determine if a student has passed in his exam or not. If the marks are greater than 40 then he has a passed else he has a failed. ◆ Determine what grade he has obtained based on the marks obtained if 90-100 is A+; 80-90 is A ;70-80 is B+; 60-70 is B ; 50-60 is C; 40-50 is D and Below 40 is fail [Hint : Use the if-else if...-else]
2.	Write a program using the switch statement to print the names of Class Teacher of all grades from 1 to 10. Hint: let the grade be a variable. Depending on its value, print the names of the class teacher.
3.	If a student has an amount greater than Rs 50 he uses an auto , else if he has an amount between Rs 20 to Rs 50 he uses the shared auto else if he has between Rs 5 to Rs 20 he uses the public bus or Else he walks to his school. Write this using the if-else if -else condition
4.	Write a JavaScript program where the program chooses a random integer between 1 to 10, then the user is prompted to input a guess number. If the user input matches with the guess number, the program will display a message "Good Work" otherwise it will display "Not matched".
5.	Write a JavaScript conditional statement to find the sign of product of three numbers. Display an alert box with the specified sign such as: <i>Sample Input numbers : 3, -7, 2</i> <i>Output : The sign is: -</i>
6.	Write a JavaScript conditional statement to sort three numbers in descending order. Display an alert box to show the result. <i>Sample Input numbers: 0, -1, 4</i> <i>Output: 4, 0, -1</i>
7.	Write a JavaScript conditional statement to find the largest of five numbers. Display an alert box to show the result. <i>Sample Input numbers : -5, -2, -6, 0, -1</i> <i>Output: 0</i>

Answer the following questions:

- Q. 1: What is the use of a conditional operator? Give some examples.
- Q. 2: Explain if the variables “name” and “Name” similar.
- Q. 3: Explain the rules to write an identifier.
- Q. 4: What do you understand by the term local scope?
- Q. 5: All students have to come to school by 9 A.M. Would this be declared as a local scope or a global scope?
- Q. 6: Will this be an acceptable string data type “Yah I’m getting through’ this way”?
- Q. 7: What is the purpose of the break statement?
- Q. 8: What is the purpose of the switch statement? Give some examples.
- Q. 9: Write a JavaScript program that accepts inputs two integers and display the larger one.
- Q. 10: Fill in the blanks
- (a) Switch statements start with keyword.
 - (b) If none of the case values in the switch statement match then is executed.
 - (c), and are all conditional statements that can be used in JavaScript
 - (d) should not be used as a variable name because it is a reserved keyword in JavaScript.
 - (e) JS has two types of variable scope &
 - (f) JavaScript variables are declared using as the keyword.
 - (g) You should not declare the same variable in the same scope

Further Readings

1. <http://www.javascriptkit.com>
2. <https://developer.mozilla.org/en/docs/JavaScript>
3. <http://www.dynamicdrive.com>

Section-4: Iterations: Loops

4.1 Loops

Loops can be used to execute the same code over and over again, particularly useful when dealing with arrays as shown in figure 3.3. JavaScript supports different kinds of loops described below:

- ◆ for - loops through a block of code a number of times
- ◆ for/in - loops through the properties of an object
- ◆ while - loops through a block of code while a specified condition is true
- ◆ do/while - also loops through a block of code while a specified condition is true

The for loop is best suited when you already know the number of times the statements should be executed. The loop executes until the condition becomes false.

Syntax:

```
for(initialization; condition; increment)
{
    //Statements    }
```

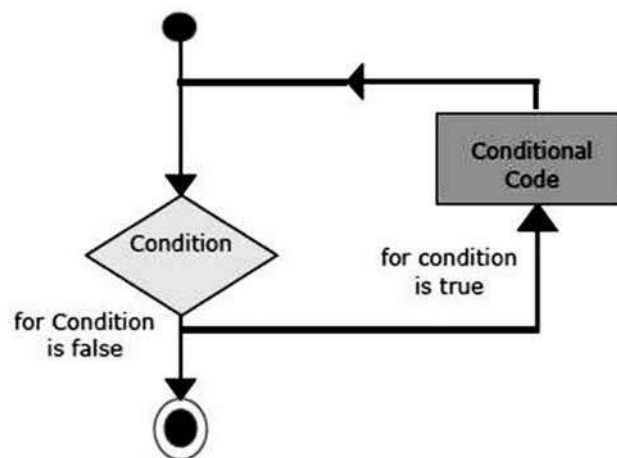


Fig. 3.3

When a **for loop** executes

1. The initializing expression is get executed and this expression usually initializes one or more loop variables.
2. The condition expression is evaluated. If the value of condition is true, the loop statements are executed. If the value of condition is false, for loop terminates.

3. The increment expression executes and increments the value by the specified step value.
4. The statements execute, and control returns to step 2.

Example

```
<!DOCTYPE html>
<html>
<body>
<p id="demo"></p>
<script>
var cars = ["BMW", "MARUTI", "HONDA", "AUDI"];
var text = "";
var i;
for (i = 0; i < cars.length; i++) {
    text += cars[i] + "<br>";
}
document.getElementById("demo").innerHTML = text;
</script>
</body>
</html>
```

Output:

BMW
MARUTI
HONDA
AUDI

Example Explained

From the above example we can notice that statement 1 sets a variable before the loop starts (var i = 0). Statement 2 defines the condition for the loop to run (i must be less than 4). Statement 3 increases a value (i++) each time the code block in the loop has been executed. In JavaScript, we normally use statement 1 to initiate the variable used in the loop (i = 0). Although, this is not always the case, JavaScript doesn't care. Statement 1 is optional. We can initiate many values in statement 1 (separated by comma):

Example

```
for (i = 0, len = cars.length, text = ""; i < len; i++)  
{  
    text += cars[i] + "<br>";  
}
```

Note: We can omit statement 1 when the values are set before the loop starts:

Example:

```
var i = 2;  
var len = cars.length;  
var text = "";  
for (; i < len; i++) {  
    text += cars[i] + "<br>";  
}
```

Statement 3 can be used for negative increment ($i--$), positive increment ($i = i + 15$), or anything else. Statement 3 can also be omitted when we increment the values inside the loop such as:

Example:

```
var i = 0;  
var len = cars.length;  
for (; i < len; ) {  
    text += cars[i] + "<br>";  
    i++;  
}
```

The for/in Loop: The JavaScript for/in statement loops through the properties of an object explain below.

Example

```
<!DOCTYPE html>
<html>
<body>
<p id="demo"></p>
<script>
var txt = "";
var person = {fname:"John", lname:"Doe", age:25};
var x;
for (x in person) {
    txt += person[x] + " ";
}
document.getElementById("demo").innerHTML = txt;
</script>
</body>
</html>
```

Output: John Doe 25

The **while** Loop:

The While Loop is another commonly used loop in JavaScript. The purpose of the while loop is to execute a block of statements over and over again until the condition fails. It is best suited in a scenario where we don't know in advanced as to how many times the loop will be executed as shown in figure 3.4.

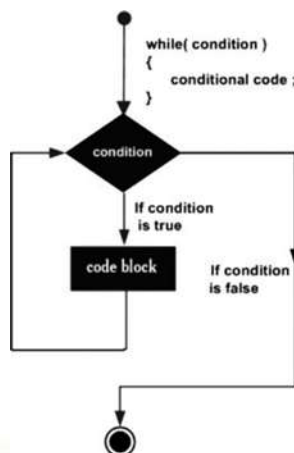


Fig. 3.4

Syntax for while loop

```
var i = 1;
while (i < 10) {
  alert(i);
  i = i + 1;
}
// At this point the value of i is 10
```

Example:

```
<!DOCTYPE html>
<html>
<body>
<p>Click the button to loop through a block of code as long as i is less than 10 </p>

<button onclick="myFunction()">Try it</button>
<p id="demo"></p>
<script>
function myFunction() {
  var text = "";
  var i = 0;
  while (i < 10) {
    text += "<br>The number is " + i;
    i++;
  }
  document.getElementById("demo").innerHTML = text;
}
</script>
</body>
</html>
```

Output: On clicking the button to Loop generates the following output on screen

```
The number is 0  
The number is 1  
The number is 2  
The number is 3  
The number is 4  
The number is 5  
The number is 6  
The number is 7  
The number is 8  
The number is 9
```

After the loop has finished, the code keeps on running from just after the closing brace (“}”) of the loop’s block.

do While Loop

This is another kind of loop and different from the for loop and the while loop. This loop will execute the statement at least once that is the statements inside the loop will always get executed at least once, even if the condition is false. The condition is checked happens after the loop has been executed. The loop will continue to execute or will terminate according to on the condition as shown in figure 3.5.

do while Syntax:

```
var i = 1;  
do {  
    alert(i);  
    i = i + 1;  
} while (i<10)  
// At this point the value of i is 10
```

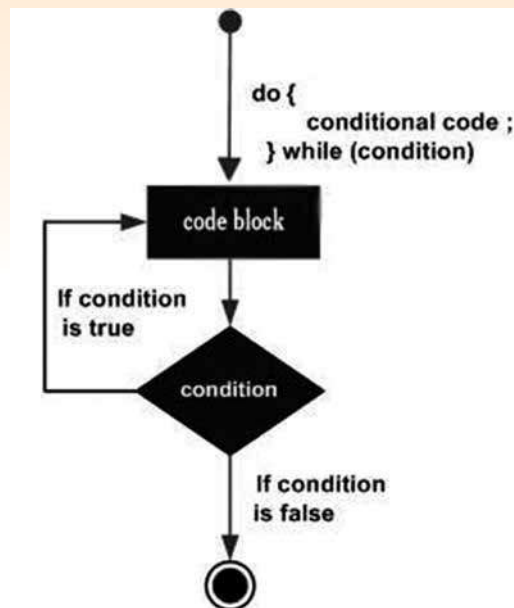


Fig. 3.5

Example:

```
<!DOCTYPE html>
<html>
<body>
<p>Click the button to loop through a block of code as long as i is less than 10.</p>
<button onclick="myFunction()">Try it</button>
<p id="demo"></p>
<script>
function myFunction() {
  var text = ""
  var i = 0;
  do {
    text += "<br>The number is " + i;
    i++;
  }
  while (i < 10)
  document.getElementById("demo").innerHTML = text; }
</script>
```

```
</body>
```

```
</html>
```

Output: On clicking the button myFunction generates the following output on screen

The number is 0

The number is 1

The number is 2

The number is 3

The number is 4

The number is 5

The number is 6

The number is 7

The number is 8

The number is 9

Break and Continue

The break statement is very useful to break the loop in the middle of an execution. The break statement can also be used to jump out of the loop. The break statement breaks the loop and continues executing the code after the loop (if any) as shown in figure 3.6.

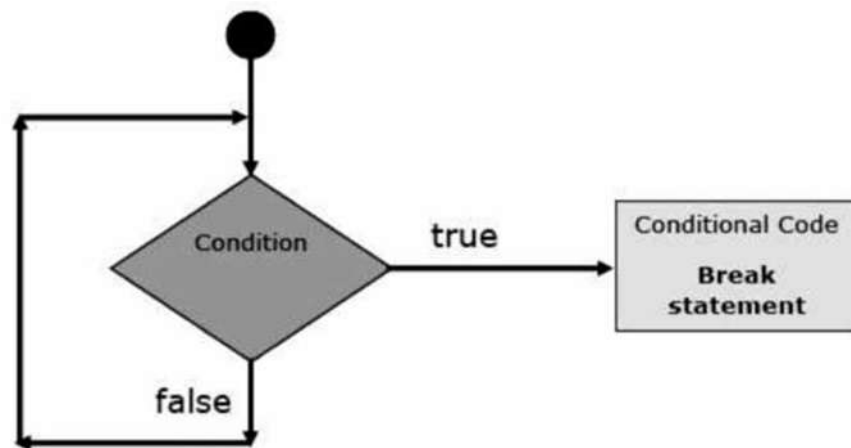


Fig. 3.6

Consider a situation where you want to check if the word 'JavaScript' is present in any of the given 10 sentences using a for loop. Suppose the computer finds it on the 3rd sentence, but keeps on running the loop unnecessarily till the end. By using the break statement here, we can exit the loop and stop the unnecessary execution.

```
<!DOCTYPE html>
<html>
<body>

<p>A loop with a break.</p>

<p id="demo"></p>

<script>
var text = "";
var i;
for (i = 0; i < 10; i++) {
    if (i === 3) { break; }
    text += "The number is " + i + "<br>";
}
document.getElementById("demo").innerHTML = text;
</script>
</body>
</html>
```

Output: This loop will stop when $i == 3$, with a break.

The number is 0

The number is 1

The number is 2

Section-4: Exercise Questions

Perform the following activities in Lab.

S. No.	Activities												
1.	Write a JavaScript program using for loop that will iterate from 0 to 15. For each iteration, it will check if the current number is odd or even, and display a message on the screen. <i>Sample Output :</i> "0 is even" "1 is odd" "2 is even" ----- -----												
2.	Write a JavaScript program which compute the average marks of the following students. <table border="1"> <thead> <tr> <th>Student Name</th><th>Marks</th></tr> </thead> <tbody> <tr> <td>Aditya</td><td>80 80</td></tr> <tr> <td>Akshat</td><td>77</td></tr> <tr> <td>Divyanshi</td><td>88</td></tr> <tr> <td>Monika</td><td>95</td></tr> <tr> <td>Aslam</td><td>68</td></tr> </tbody> </table>	Student Name	Marks	Aditya	80 80	Akshat	77	Divyanshi	88	Monika	95	Aslam	68
Student Name	Marks												
Aditya	80 80												
Akshat	77												
Divyanshi	88												
Monika	95												
Aslam	68												
3.	Write a JavaScript program to find the armstrong number of 3 digits. Note : An Armstrong number of three digits is an integer such that the sum of the cubes of its digits is equal to the number itself. For example, 371 is an Armstrong number since $3^3 + 7^3 + 1^3 = 371$.												
4.	Write a JavaScript program to dising the following pattern, using a nested for loop. <pre> * * * * * * * * * * * * * * *</pre>												

5.	Write a JavaScript program to calculate the factorial of a number. In mathematics, the factorial of a non-negative integer n , denoted by $n!$, is the product of all positive integers less than or equal to n . For example, $5! = 5 \times 4 \times 3 \times 2 \times 1 = 120$
6.	Write a JavaScript program to get the first n Fibonacci numbers. Note : The Fibonacci Sequence is the series of numbers: 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, . . Each subsequent number is the sum of the previous two numbers.
7.	Write a sorting program in JavaScript for ascending and descending orders. Sample Input numbers: 34, 7, 23, 32, 5, 62 Sample output: 5, 7, 23, 32, 34, 62 (Ascending Order)

Answer the following questions:

- Q. 1: What is a loop?
- Q. 2: What are the three main types of loop?
- Q. 3: Differentiate between do-while and while-do loop.
- Q. 4: Explain the purpose of loops in JavaScript.
- Q. 5: Explain the while and do-while loops and when to use each of them.
- Q. 6: Explain a real world situation where the break statement helps.
- Q. 7: Explain where the continue statement could help.
- Q. 8: Write a JavaScript program to compute the greatest common divisor (GCD) of two positive integers.
- Q. 9: Write a JavaScript program to sum the multiples of 3 and 5 under 1000.
- Q. 10: Fill in the blanks
- execute the same set of operations over and over again.
 - The executes at least once irrespective of the condition.
 - statement skips the loop execution for a particular iteration.
 - The statement exits the loop and stops unnecessary execution of the loop.
 - Difference between do-while loop and while loop is

Further Readings

1. <http://www.javascriptkit.com>
2. <https://developer.mozilla.org/en/docs/JavaScript>
3. <http://www.dynamicdrive.com>

Section-5: Window Object

The window object represents an open window in a browser. If a document contain frames (<iframe> tags), the browser creates one window object for the HTML document, and one additional window object for each frame. There is no public standard that applies to the Window object, but all major browsers support it.

5.1 Window Object Properties

Properties	Description
closed	Returns the Boolean variable indicating whether the window has been closed or not.
defaultStatus	Read/write property that reflects the default window status bar message that appears.
document	Returns the document object reference to the current document object.
frames	Returns all <iframe> elements in the current window. An array referencing all of the frames in the current window, including IFRAME elements. If the current page is contained in a frame itself and you wish to access another sibling frame, navigate up towards the topmost frameset document first using the parent property of window, for example: <code>parent.frames[0]</code> //access first frame within parent frameset.
history	Reference to the History object of JavaScript, which contains information on the URLs, the visitor has visited within the window.

Properties	Description
innerWidth, innerHeight - <i>Supported in: Firefox/Opera/ Safari, but NOT IE.</i>	Returns the width and height, in pixels, of the window's viewable content area, which does not include any toolbar, scrollbars etc. innerWidth returns the inner width of a window's content area and innerHeight returns the inner height of a window's content area
outerWidth, outerHeight - <i>Supported in: Firefox/Opera/ Safari, but NOT IE.</i>	Returns the width and height, in pixels, of the browser window in its entirety, which includes toolbar, scrollbars etc. No IE equivalents.
length	Returns the number of frames contained in the window, which includes IFRAMEs.
location	Returns or sets the location object of JavaScript, which contains information on the current URL.
name	The name of the window as optionally specified when calling window.open().
opener	Contains a reference to the window that opened the secondary window via window.open(). This property should be invoked in the secondary window.
parent	Reference to the parent frameset window of the current window, assuming that current window is a frame. Otherwise, it simply refers to current window.
self	A synonym for the current window.
status	A read/write property that allows you to probe and write to the browser's status bar.
top	A synonym for the topmost browser window.
window	References the current window. Same as "self."

Properties	Description
pageXOffset, pageYOffset <i>- Supported in: Firefox/Opera/ Safari, but NOT IE.</i>	Returns an integer representing the pixels the current document has been scrolled from the upper left corner of the window, horizontally and vertically, respectively. You can also use window.scrollX and window.scrollY.
screenX, screenY <i>- Supported in: Firefox/Opera/ Safari, but NOT IE.</i>	Specifies the x and y coordinates of the window relative to the user's monitor screen. Note: IE+ equivalents are "window.screenLeft" and "window.screenTop"
screenLeft, screenTop <i>- IE only properties</i>	Specifies the x and y coordinates of the window relative to the user's monitor screen.

5.2 Window Object Methods

In the following table "[]" surrounding a parameter within the square brackets means the parameter is optional. You can ignore it.

Methods	Description
alert(msg)	Displays an Alert dialog box with the desired message and OK button.
blur()	Removes focus from the window in question, sending the window to the background on the user's desktop.
clearInterval(ID)	Clears the timer set using var ID=setInterval().
clearTimeout(ID)	Clears the timer set using var ID=setTimeout().
close()	Closes a window.
confirm(msg)	Displays a Confirm dialog box with the specified message and OK and Cancel buttons. Returns either true or false, depending on the button, that the user has clicked on, for example: <pre>var yourstate=window.confirm("Are you sure you are ok?") if (yourstate) //Boolean variable. Sets to true if user pressed "OK" versus "Cancel." window.alert("Good!")</pre>

Methods	Description
find(string, [casesensitive], [backward]) - Firefox only property	Searches for the “string” within the page, and returns string or false, accordingly. “casesensitive” is a Boolean denoting whether search is case sensitive. “backwards” is a Boolean which when set to true, searches the page backwards. Final two optional parameters must be set together or none at all.
focus()	Sets focus to the window, bringing it to the forefront on the desktop.
home() - Firefox only property	Navigates the window to the homepage as designated by the user’s browser setting.
moveBy(dx, dy)	Moves a window by the specified amount in pixels.
moveTo(x, y)	Moves a window to the specified coordinate values, in pixels. The following opens a window and centers it on the user’s screen
open(URL, [name], [features], [replace])	Opens a new browser window. “Name” argument specifies a name that you can use in the target attribute of your <a> tag. “Features” allows you to show/hide various aspects of the window interface. “Replace” is a Boolean argument that denotes whether the URL loaded into the new window should add to the window’s history list.
print()	Prints the contents of the window or frame.
prompt(msg, [input])	Displays a Prompt dialog box with a message. Optional “input” argument allows you to specify the default input (response) that gets entered into the dialog box. This function returns the string the user has entered such as: var yourname=window.prompt(“please enter your name”) alert(yourname)

Methods	Description
resizeBy(dx, dy)	Resizes a window by the specified amount in pixels.
resizeTo(x y)	Resizes a window to the specified pixel values. The following will resize the current window to be maximized on the screen in browsers with no security hang ups over such an operation (IE does and will do nothing): window.resizeTo(screen.availWidth, screen.availHeight)
scrollBy(dx, dy)	Scrolls a window by the specified amount in pixels.
scrollTo(x, y)	Scrolls a window to the specified pixel values.
setTimeout("func", interval, [args])	Calls the specified function reference (func) or JavaScript statement(s) once after the "interval" parameter has expired, in milliseconds (ie: 1000=after 1 second). This method returns a unique ID which can be passed into clearTimeout(id) to clear the timer.
stop()	Stops the window from loading. NS4/NS6+ exclusive method. Use the optional "args" to pass any number of arguments to the function.

5.3 Window Object Events

Events	Description
onblur	Fires when the window loses focus.
onerror	Fires when a JavaScript error occurs. By returning true inside this event, JavaScript errors on the page (if any) are suppressed, with no error messages popping up.
onfocus	Fires when the focus is set on the current window.
onload	Fires when the page has finished loading, including images. This is a popular event to use to run some JavaScript once everything on the page has loaded/ is available: window.onload=function(){ runsomefunction() }
onresize	Fires when the window is resized.

Events	Description
onscroll	Fires when the window is scrolled. The following shows the current y coordinate of the upper left corner of the viewable window in the browser's title bar when the page is scrolled: <pre> window.onscroll=function(){ var scrollY=window.pageYOffset document.body.scrollTop document.title=scrollY } </pre>
onbeforeunload	Fires when the page is about to be unloaded, prior to window.onunload event firing. Supported in all modern browsers. By setting event.returnValue to a string, the browser will prompt the user whether he/she wants to leave the current page when attempting to: <pre> window.onbeforeunload=function(e){ e.returnValue="Any return string here forces a dialog to appear when user leaves this page" } </pre> window.location="http://www.google.com" //prompt is invoked
onunload	Fires when the page is unloaded- process cannot be overruled at this point. Often used to run code cleanup routines.

Window Features in window.open

There are several features you can pass into the "feature" parameter of **window.open()** to manipulate its interface. Most features support a value of true or false. Though, in general, simply including the name of the feature implies it should be added to the window (yes), while not including it means it shouldn't (no). You should apply each feature separated with a comma (,).

Feature	Description
channelmode	Specifies if window should be opened in channel mode. IE only.
fullscreen	Specifies if window should be opened in full screen mode. IE only.
height	Specifies the height of the window.
left	Specifies the x coordinates of the window in pixels. IE only.
location	Specifies if the location bar of the window should be included.

Feature	Description
menubar	Specifies if the menu bar of the window should be included.
resizable	Specifies if window should be resizable.
screenX	Specifies the x coordinates of the window in pixels. NS only. See “left” as well.
screenY	Specifies the x coordinates of the window in pixels. NS only. See “top” as well.
scrollbars	Specifies if window should contain scrollbars.
status	Specifies if the status bar of the window should be included.
toolbar	Specifies if the toolbar of the window (ie: reload button) should be included.
top	Specifies the y coordinates of the window in pixels. IE only. See “screenY” as well.
width	Specifies the width of the window.

5.4 Location Object

The location object contains information about the current URL. The location object is part of the window object and is accessed through the window location property.

Note: There is no public standard that applies to the location object, but all major browsers support it.

Location Object Properties

Property	Description
hash	Sets or returns the anchor part (#) of a URL
host	Sets or returns the hostname and port number of a URL
hostname	Sets or returns the hostname of a URL
href	Sets or returns the entire URL
origin	Returns the protocol, hostname and port number of a URL
pathname	Sets or returns the path name of a URL
port	Sets or returns the port number of a URL
protocol	Sets or returns the protocol of a URL
search	Sets or returns the query string part of a URL

5.5 Location Object Methods

Method	Description
assign()	Loads a new document
reload()	Reloads the current document
replace()	Replaces the current document with a new one

5.6 History Object

The history object contains the URLs visited by the user (within a browser window). The history object is a part of the window object and is accessed through the window.history property.

Note: There is no public standard that applies to the history object, but all major browsers support it.

History Object Properties

Property	Description
length	Returns the number of URLs in the history list

History Object Methods

Method	Description
back()	Loads the previous URL in the history list
forward()	Loads the next URL in the history list
go()	Loads a specific URL from the history list

Section-5: Exercise Questions

Perform the following activities in Lab.

S. No.	Activities
1.	Write a Javascript program using onerror event when a JavaScript error occurs. By returning true inside this event, JavaScript errors on the page (if any) are suppressed, with no error messages popping up.
2.	Write a Javascript program using onload event when the page has finished loading, including images. Note: This is a popular event to use to run some JavaScript once everything on the page has loaded/ is available.
3.	Write a Javascript program using frames.length to probe the number of frames.

S. No.	Activities
4.	Write a Javascript program where an integer representing the pixels the current document has been scrolled from the upper left corner of the window, horizontally and vertically, respectively. You can also use window.scrollX and window.scrollY instead for brevity, which are equivalent properties. Hint: Apply pageXOffset, pageYOffset properties of window objects.
5.	Write Javascript for displaying an Alert dialog box with the desired message and OK button.
6.	Write a Javascript program for moving a window to the specified coordinate values, in terms of pixels and also opens a window and centers it on the user's screen.

Answer the following questions:

- Q. 1: What are different types of window Objects?
- Q. 2: Explain any three features of Window of window.open.
- Q. 3: What is an event?
- Q. 4: Write any three location object properties.
- Q. 5: Define location object methods.
- Q. 6: Write any two location object methods.
- Q. 7: Define any history object.
- Q. 8: Write any property of history object.
- Q. 9: Explain back() and forward() method of history object.
- Q. 10: What is onerror event?
- Q. 11: What is onfocus event?
- Q. 12: What is onload event?
- Q. 13: Differentiate between innerWidth and innerHeight properties of the Window.
- Q. 14: What is length property of the Window object.
- Q. 15: Write parent property of the Window object.
- Q. 16: Fill in the blanks:
- (a) Event fires when the window is resized.
- (b) Event fires when the window is scrolled.

- (c) Event fires when the page is about to be unloaded, **prior** to window.onunload event firing.
- (d) property of the window returns the Boolean variable indicating whether window has been closed or not
- (e) property of the window that reflects the default window status bar message that appears.
- (f) property of the window that references to the History object of JavaScript, which contains information on the URLs the visitor has visited within the window.
- (g) property of the window that returns the width and height, in pixels, of the window's viewable content area, which does not include any toolbar, scrollbars etc
- (h) property of the window returns or sets the location object of JavaScript, which contains information on the current URL
- (i) property of the window that returns an integer representing the pixels the current document has been scrolled from the upper left corner of the window, horizontally and vertically, respectively.
- (j) The window method which removes focus from the window in question, sending the window to the background on the user's desktop
- (k) The window method which displays a Confirm dialog box with the specified message and OK and Cancel buttons. Returns either true or false, depending on which button the user has clicked, is
- (l) The window method which sets focus to the window, bringing it to the forefront on the desktop is
- (m) The window method which displays a Prompt dialog box with a message is
- (n) The window feature in window.open which specifies the x coordinates of the window in pixels is
- (o) The window feature in window.open which specifies the y coordinates of the window in pixels is

Further Readings

<http://www.javascriptkit.com>

<https://developer.mozilla.org/en/docs/JavaScript>

<http://www.dynamicdrive.com>

Section-6: Popup Boxes

Popup Boxes

JavaScript has three kind of popup boxes: Alert box, Confirm box, and Prompt box.

6.1 Alert Box

What is an alert Box?

An alert box is often used if you want to make sure information comes through to the user.

When an alert box pops up, the user will have to click “OK” to proceed.

Syntax

```
window.alert("sometext");
```

The **window.alert()** method can be written without the window prefix.

Example

```
alert("I am an alert box!");
```

Try It Out!

```
<!DOCTYPE html>
<html>
<body>
<p>Click the button to display an alert box:</p>
<button onclick="myFunction()">Try it</button>
<script>
function myFunction() {
    alert("I am an alert box!");
}
</script>
</body>
</html>
```

6.2 Confirm Box

A confirm box is often used if you want the user to verify or accept something. When a confirm box pops up, the user will have to click either “OK” or “Cancel” to proceed further. If the user clicks “OK”, the box returns true. If the user clicks “Cancel”, the box returns false.

Syntax

```
window.confirm(sometext);
```

The **window.confirm()** method can be written without the window prefix.

Example

```
var r = confirm("Press a button");
if (r == true) {
    x = "You pressed OK!";
} else {
    x = "You pressed Cancel!";
}
```

Try It Out!

```
<!DOCTYPE html>
<html>
<body>
<p>Click the button to display a confirm box.</p>
<button onclick="myFunction()">Try it</button>
<p id="demo"></p>
<script>
function myFunction() {
    var x;
    if (confirm("Press a button!") == true) {
        x = "You pressed OK!";
    } else {
        x = "You pressed Cancel!";
    }
    document.getElementById("demo").innerHTML = x;
}
</script>
</body>
</html>
```

6.3 Prompt Box

A prompt box is often used if you want the user to input a value before entering a page.

When a prompt box pops up, the user will have to click either “OK” or “Cancel” to proceed after entering an input value. If the user clicks “OK” the box returns the input value. If the user clicks “Cancel” the box returns null.

Syntax

```
window.prompt(sometext, defaultText);
```

The **window.prompt()** method can be written without the window prefix.

Example:

```
var person = prompt("Please enter your name", "Harry Potter");
if (person != null) {
    document.getElementById("demo").innerHTML =
        "Hello " + person + "! How are you today?";
}
```

Try It Out!

```
<!DOCTYPE html>
<html>
<body>

<p>Click the button to demonstrate the prompt box.</p>
<button onclick="myFunction()">Try it</button>
<p id="demo"></p>
<script>
function myFunction() {
    var person = prompt("Please enter your name", "Harry Potter");

    if (person != null) {
        document.getElementById("demo").innerHTML =
            "Hello " + person + "! How are you today?";
    }
}
</script>
</body>
</html>
```

Section-6: Exercise Questions

Appendix A: Exercises for practice using JavaScript

1. Check the output generated from the following code and analyze the result

```
<!DOCTYPE html>
<html>
<body>
<h1>My First JavaScript</h1>
<button type="button"
onclick="document.getElementById('demo').innerHTML = Date()">
Click me to display Date and Time.</button>
<p id="demo"></p>
</body>
</html>
```

2. Check the output generated from the following code and analyze the result

```
<!DOCTYPE html>
<html>
<body>
<p id="demo">Display the result here.</p>
<script>
// Create the array here
</script>
</body>
</html>
```

3. Check the output generated from the following code and analyze the result

```
<!DOCTYPE html>
<html>
<body>
<p id="demo"></p>
<script>
if () {
    document.getElementById("demo").innerHTML = "Well done!";
}
</script>
</body>
```

```
</html>
```

4. Check the output generated from the following code and analyze the result

```
<!DOCTYPE html>
<html>
<body>
<p id="demo">Display the result here.</p>
<script>
var firstName = "Greg";
if (firstName === "John") {
    document.getElementById("demo").innerHTML = "Hello John!";
}
</script>
</body>
</html>
```

5. Check the output generated from the following code and analyze the result

```
<!DOCTYPE html>
<html>
<body>
<p id="demo"></p>
<script>
var i;
for (i = num1; i < num2; i++) {
    document.getElementById("demo").innerHTML += i + "<br>";
}
</script>
</body>
</html>
```

6. Check the output generated from the following code and analyze the result

```
<!DOCTYPE html>
<html>
<body>
<p id="demo"></p>
<script>
var text = "";
var food = ["Pizza", "Beans", "Tacos", "Fish", "Chicken"];
```

```
var i;  
for () {  
    text += "I love " + food[i] + "<br>";  
}  
document.getElementById("demo").innerHTML = text;  
</script>  
</body>  
</html>
```

7. Check the output generated from the following code and analyze the result

```
<!DOCTYPE html>  
<html>  
<body>  
<p id="demo"></p>  
<script>  
function myFunction() {  
    document.getElementById("demo").innerHTML = "Hello World!";  
}  
// Call the function here  
</script>  
</body>  
</html>
```

8. Check the output generated from the following code and analyze the result

```
<!DOCTYPE html>  
<html>  
<body>  
<p id="demo"></p>  
<script>  
function myFunction() {  
    // Add code here  
}  
document.getElementById("demo").innerHTML = myFunction();  
</script>  
</body>
```

```
</html>
```

9. Check the output generated from the following code and analyze the result

```
<!DOCTYPE html>
<html>
<body>
<p id="demo">Display the result here.</p>
<script>
// Create the variables here
document.getElementById("demo").innerHTML =
firstName + " " + lastName + " is " + age;
</script>
</body>
</html>
```

10. Check the output generated from the following code and analyze the result

```
<!DOCTYPE html>
<html>
<body>
<p id="demo">Display the result here.</p>
<script>
var carName = "Volvo";
document.getElementById("demo").innerHTML = carname;
</script>
</body>
</html>
```

Answer the following questions:

- Q. 1: What are major types of Javascript popup boxes?
 Q. 2: Define an alert Box.
 Q. 3: What is the purpose of defining alert Box?
 Q. 4: Define a confirm Box.
 Q. 5: What is the purpose of defining confirm Box?

Q. 6: Define an prompt Box.

Q. 7: What is the purpose of defining prompt Box?

Q. 8: Fill in the blanks

- (a) When a prompt box pops up, the user will have to click to proceed after entering an input value.
- (b) When an alert box pops up, the user will have to click to proceed after entering an input value.
- (c) When a confirm box pops up, the user will have to click to proceed after entering an input value.
- (d) When a prompt box pops up, the user will have to click to proceed after entering an input value.

Further Readings

1. <http://www.javascriptkit.com>
2. <https://developer.mozilla.org/en/docs/JavaScript>
3. <http://www.dynamicdrive.com>

Section-7: Arrays

7.1 Arrays

JavaScript arrays are used to store multiple values in a single variable. Arrays in JavaScript are actually a special type of object. They work similar to regular objects but they have one special property called 'length'. The length of the array (size of the array) is always one more than the highest index in the array. The traditional way of creating arrays is as follows:

Syntax:

```
var a = new Array();  
a[0] = "dog";  
a[1] = "cat";  
a[2] = "hen";  
a.length
```

Output: 3

A more convenient notation can be used while dealing with arrays.

Example:

```
var a = ["Apple", "Mangoes", "Orange", "Banana"];  
a.length
```

Output: 4

Note: Leaving a trailing comma at the end of an array literal is inconsistent across browsers, so don't do it. The length of the array is one more than the highest index. However, the array.length isn't necessarily the number of items in the array.

Example:

```
var a = ["dog", "cat", "hen"];  
a[100] = "fox";  
a.length
```

Output: 101

If we query a non-existent array index, we get undefined result shown as below.

Example: `typeof a[90]`

Output: undefined

If we take the above example into account, we can iterate over an array using the following:

```
for(var i = 0; i < a.length; i++)  
{  
    // Do something with a[i]  
}
```

Example:

```
<!DOCTYPE html>
<html>
<body>
<p id="demo"></p>
<script>
var cars = ["Audi", "Volvo", "BMW"];
document.getElementById("demo").innerHTML = cars[0];
</script>
</body>
</html>
```

Output: Audi

```
<!DOCTYPE html>
<html>
<body>
<p id="demo"></p>
<script>
var cars = [
    "Audi",
    "Volvo",
    "BMW" ];
document.getElementById("demo").innerHTML = cars[0];
</script>
</body>
</html>
```

Output: Audi

Arrays Using the JavaScript Keyword new

```
var cars = new Array("Audi", "Maruti", "BMW");
```

```
<!DOCTYPE html>
<html>
<body>

<p id="demo"></p>

<script>
var cars = new Array("Saab", "Volvo", "BMW");
document.getElementById("demo").innerHTML = cars[0];
</script>

</body>
</html>
```

Output: Audi

7.2 Arrays & Objects

Arrays are a special type of objects. The `typeof` operator in JavaScript returns "object" for arrays. Arrays use numbers to access its elements. Consider the following example:

```
<!DOCTYPE html>
<html>
<body>

<p id="demo"></p>

<script>
var person = ["John", "Doe", 46];
document.getElementById("demo").innerHTML = person[0];
</script>

</body>
</html>
```

Output: In this example `person[0]` returns John

Objects can also use names to access its members. In the above example, `person.firstName` returns John. We can declare the object alternatively such as:

```
var person = {firstName:"John", lastName:"Doe", age:46};
```

The strength of JavaScript arrays are the built-in array properties and methods:

Examples

```
var x = cars.length;      // The length property returns the number of elements in cars
var y = cars.sort();      // The sort() method sort cars in alphabetical order
```

The length Property

The length property of an array returns the length of an array (the number of array elements).

Example

```
var fruits = ["Banana", "Orange", "Apple", "Mango"];
fruits.length;      // the length of fruits is 4
```

Adding Array Elements

The easiest way to add a new element to an array is using the push method:

Example

```
var fruits = ["Banana", "Orange", "Apple", "Mango"];
fruits.push("Lemon"); // adds a new element (Lemon) to fruits
```

```
<!DOCTYPE html>
<html>
<body>

<p>The push method appends a new element to an array.</p>
<button onclick="myFunction()">Try it</button>
<p id="demo"></p>
<script>
var fruits = ["Banana", "Orange", "Apple", "Mango"];
document.getElementById("demo").innerHTML = fruits;

function myFunction() {
  fruits.push("Lemon")
  document.getElementById("demo").innerHTML = fruits;
}
</script>
</body>
</html>
```

Output: The push method appends a new element to an existing array every time you execute the program.

New element can also be added to an array using the length property:

Example

```
var fruits = ["Banana", "Orange", "Apple", "Mango"];  
fruits[fruits.length] = "Lemon";    // adds a new element (Lemon) to fruits
```

Looping Array Elements

The best way to loop through an array, is using with the help of for loop:

Example

```
<!DOCTYPE html>  
<html>  
<body>  
  
<p>The best way to loop through an array is using a standard for loop:</p>  
  
<button onclick="myFunction()">Try it</button>  
  
<p id="demo"></p>  
  
<script>  
function myFunction() {  
    var index;  
    var text = "<ul>";  
    var fruits = ["Banana", "Orange", "Apple", "Mango"];  
    for (index = 0; index < fruits.length; index++) {  
        text += "<li>" + fruits[index] + "</li>";  
    }  
    text += "</ul>";  
    document.getElementById("demo").innerHTML = text;  
}  
</script>  
  
</body>  
</html>
```

7.3 Objects

JavaScript objects are simply collections of name-value pairs. The “name” part is a JavaScript string, while the value can be any JavaScript value including more objects. There are two basic ways to create an empty object:

```
var obj = new Object( ); and  
var obj = { };
```

These are semantically equivalent, the second is called object literal syntax, and is more convenient. Once created, an object’s properties can again be accessed in one of two ways:

```
obj.name = “Simon”;  
var name = obj.name;  
obj[“name”] = “Simon”;  
var name = obj[“name”];
```

These are also semantically equivalent. The second method has the advantage that the name of the property is provided as a string, which means it can be calculated at run-time. It can also be used to set and get properties with names that are reserved words:

```
obj.for = “Simon”;           // Syntax error, because ‘for’ is a reserved word  
obj[“for”] = “Simon”;        // works fine
```

Object literal syntax can be used to initialise an object :

```
var obj = {  
    name: “Carrot”,  
    “for”: “Max”,  
    details: {  
        color: “orange”,  
        size: 12  
    }  
}
```

Attribute access can be chained together:

To find the color of the given object

```
obj.details.color           //output orange
```

To get the size of the given object

```
obj["details"]["size"] //output is 12
```

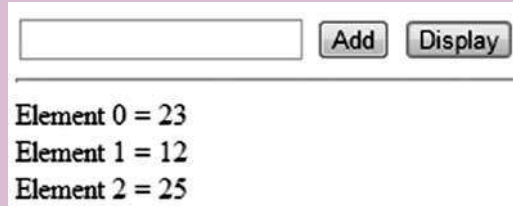
Arrays and string come with a number of methods summarized as follows:

substring()	Extracts the characters from a string, between two specified indices
concat()	Joins two or more strings, and returns a copy to the joined strings
join()	Joins all elements of an array into a string
pop()	Removes and returns the last item.
push()	Push adds one or more items to the end.
reverse()	Reverses the order of an element in an array
shift()	Removes the first element of an array and returns that element
slice()	Returns a sub-array.
sort()	Takes an optional comparison function.
splice()	Modify an array by deleting a section and replacing it with more items.
unshift()	Prep ends items to the start of the array.

Section-7: Exercise Questions

Perform the following activities in Lab:

S. No.	Activities
1.	Write a simple JavaScript program to join all elements of the following array into a string. <i>Sample Input array</i> : myColor = ["Red", "Green", "White", "Black"]; <i>Expected Output</i> : "Red,Green,White,Black" "Red,Green,White,Black" "Red+Green+White+Black"
2.	Write a JavaScript program which accepts a number as input and insert dashes (-) between each two even numbers. For example if you accept 025468 the output should be 0-254-6-8.

S. No.	Activities
3.	Write a JavaScript program to sort the items of a given array in ascending order. <i>Sample Input array</i> : <code>var arr1 = [3, 8, 7, 6, 5, -4, 3, 2, 1];</code> <i>Sample Output</i> : -4,-3,1,2,3,5,6,7,8
4.	Write a JavaScript program to add items in an blank array and display the items. <i>Sample Input Screen</i> : 
5.	Write a JavaScript program which prints the elements of the following array. Note : Use nested for loops. <i>Sample Input array</i> : <code>var a = [[1, 2, 1, 24], [8, 11, 9, 4], [7, 0, 7, 27], [7, 4, 28, 14], [3, 10, 26, 7]];</code> <i>Sample Output</i> : "row 0" " 1" " 2" " 1" " 24" "row 1" -----

Answer the following questions:

- Q. 1: What is an Array? Explain the purpose of defining array.
- Q. 2: What are major types of arrays?
- Q. 3: Explain the procedure to access an array.
- Q. 4: Write a JavaScript program to find the most frequent item of an array.
Sample Input array : `var arr1=[3, 'a', 'a', 'a', 2, 3, 'a', 3, 'a', 2, 4, 9, 3];`
Sample Output : a (5 times)
- Q. 5: Write a JavaScript program which accept a string as input and swap the case of each character. For example if you input 'The Quick Brown Fox' the output should be 'tHE qUICK bROWN fOX'.

- Q. 6: Write a JavaScript program to find the sum of squares of a numeric given array.
- Q. 7: Write a JavaScript program to compute the sum and product of an array of integers.
- Q. 8: Write a JavaScript program to remove duplicate items from an array (ignore case sensitivity).
- Q. 9: How will you define an object in Javascript?
- Q. 10: Fill in the blanks
- (a) An array is accessed by the numbers.
 - (b) Property is used to find the number of elements in an array.
 - (c) joins two or more strings, and returns a copy to the joined string
 - (d) extracts the characters from a string, between two specified indices
 - (e) joins all elements of an array into a strings
 - (f) removes and returns the last item
 - (g) reverses the order of an element in an array

Further Readings

1. <http://www.javascriptkit.com>
2. <https://developer.mozilla.org/en/docs/JavaScript>
3. <http://www.dynamicdrive.com>

Section-8: Functions

Functions

Functions are reusable blocks of code that carry out a specific task. To execute the code in a function we need to call it. A function can be passed through arguments to use, and a function may return a value to whatever called it. You can save functions as the value of a variable, and you call a function using this variable and a pair of parentheses. This is also called invoking the function.

A Function may be defined as a piece of code that does not execute until you specifically call it. The code can be reused in many places across files. This saves a lot of code from been re-written unnecessarily. Depending on the need you can write the function on your own or reuse functions written by other professional programmers.

Function Syntax

A JavaScript function is defined with the 'function' keyword, followed by a name, followed by parentheses (). Function names can contain letters, digits, underscores, and dollar signs (similar to the rules for declaring as variables). The parentheses may include parameter names separated by commas: (*parameter1*, *parameter2*, ...). The code to be executed, by the function, is placed inside curly brackets '{}'

Function Declaration

```
function name (parameter1, parameter2, parameter3)  
{  
    code to be executed  
}
```

Function parameters are the names listed in the function definition and function arguments are the real values received by the function when it is invoked. Inside the function, the arguments are used as local variables.

To create a function, use the 'function' keyword. You can then list the arguments in parentheses, and then supply a block that contains the function's code. Here's a function that adds two numbers:

```
var add = function (a, b)  
{  
    return a + b;  
};
```

a and b are the two function's parameters, and the value it returns is signified by the return keyword. The return keyword also stops execution of the code in the function.

```
var result = add(1, 2); // it will generate result 3
```

This calls function add with the arguments 1 and 2, which will be saved in the variables a and b.

A JavaScript function can take 0 or more named parameters. The function body can contain as many statements and can declare its own variables which are local to that function. The return statement can be used to return a value at any time, or terminating the function. If no return statement is used (or an empty return with no value), JavaScript returns undefined.

Function Invocation

The code inside the function will execute when “something” invokes (calls) the function:

- ◆ When an event occurs (when a user clicks a button)
- ◆ When it is invoked (called) from JavaScript code
- ◆ (self invoked) when it is invoked automatically.

Function Return

When JavaScript reaches a return statement, the function will stop executing. If the function was invoked from a statement, JavaScript will “return” to execute the code after the invoking statement. Functions often compute a return value. The return value is “returned” back to the “caller”:

Note: JavaScript allows you to return a single value from a function. The return statement must be the last line in the function.

Example: Calculate the product of two numbers, and return the result:

```
<!DOCTYPE html>
<html>
<body>
<p>This example calls a function which performs a calculation, and returns the result:</p>
<p id="demo"></p>
<script>
function myFunction(a, b) {
    return a * b;
}
document.getElementById("demo").innerHTML = myFunction(4, 3);
</script>
</body>
</html>
```

Output: This example calls a function “**myFunction**” which performs a calculation, and returns the result:12

Why there is need for Functions?

You can reuse code, define the code once, and use it many times. You can use the same code many times with different arguments, to produce different results.

Example: Convert Fahrenheit to Celsius:

```
<!DOCTYPE html>
<html>
<body>
<p>This example calls a function to convert from Fahrenheit to Celsius:</p>
<p id="demo"></p>
<script>
function toCelsius(f) {
    return (5/9) * (f-32);
}
document.getElementById("demo").innerHTML = toCelsius(32);
</script>
</body>
</html>
```

Output: This example calls a function “toCelsius” to convert from Fahrenheit to Celsius, result is 0

The () Operator Invokes the Function

Using the example above, toCelsius refers to the function object, and toCelsius() refers to the function result.

Example: Accessing a function without () will return the function definition:

```

<!DOCTYPE html>
<html>
<body>
<p>Accessing a function without (), will return the function definition:</p>
<p id="demo"></p>
<script>
function toCelsius(f) {
    return (5/9) * (f-32);
}
document.getElementById("demo").innerHTML = toCelsius;
//document.getElementById("demo").innerHTML = toCelsius(32);
</script>
</body>
</html>

```

Output: Accessing a function without (), will return the function definition like, function toCelsius(f) { return (5/9) * (f-32); }

Functions Used as Variables: In JavaScript, you can use functions the same way as you use variables.

Example: Using JavaScript functions the same way as you have used variables

```

<!DOCTYPE html>
<html>
<body>
<p id="demo"></p>
<script>
document.getElementById("demo").innerHTML =
"The temperature is " + toCelsius(32) + " Centigrade";

function toCelsius(fahrenheit) {
    return (5/9) * (fahrenheit-32);
}
</script>
</body>
</html>

```

Output: The temperature is 0 Centigrade

Section-8: Exercise Questions

Perform the following activities in Lab.

S. No.	Activities
1.	Create a program using functions to: a. Pass 3 arguments a, b and c to a function. Add the values a & b. Multiply c with the sum of a & b. b. Pass your name as an argument and display “Hello My Name is <yourname>”
2.	Write a program and pass a number to a function and determine if the number is greater than 50. Return true if it is greater and false if it not.
3.	Write a program using a function named calculateAge which: takes 2 arguments: birth year and current year. calculates the possible ages based on those years. outputs the result to the screen like so: “You are either “NN or NN” Call the function three times with different sets of values.
4.	Write a program using function called celsiusToFahrenheit that: Store a celsius temperature into a variable. Convert it to fahrenheit and output “NN°C is NN°F”. Create a function called fahrenheitToCelsius: Now store a fahrenheit temperature into a variable. Convert it to celsius and output “NN°F is NN°C.”
5.	Write a JavaScript function that checks whether a (passed) string enter by the user is palindrome or not? A palindrome is word, phrase, or sequence that reads the same backward as forward, e.g., madam or nurses run.

Answer the following questions:

- Q. 1: Explain the purpose of a function.
- Q. 2: Explain the use of a return statement.
- Q. 3: What do you understand by local and global variables? Explain using a function specifying the value of the local variable and the global variable.
- Q. 4: Write a JavaScript program using function to get the current date. Expected Output: mm-dd-yyyy, mm/dd/yyyy or dd-mm-yyyy, dd/mm/yyyy.

- Q. 5: Write a JavaScript program using function to find the area of a triangle where lengths of the three of its sides are 5, 6, 7.
- Q. 6: Write a JavaScript function that reverses a number. For example $x = 32243$; then Expected Output : 34223
- Q. 7: Write a JavaScript function to compute the factors of a positive integer.
- Q. 8: Write a JavaScript program to calculate multiplication and division of two numbers (input from the user) using 2 functions namely Multiply and Divide shown as below:

Sample Output form :

1st Number :

2nd Number:

The Result Is :

120

- Q. 9: Fill in the blanks
- (a) The function is used to display the current day and time
 - (b) A is not executed until you call it.
 - (c) The Return statement returns a value to the calling statement.
 - (d) A function should always return
 - (e) A function is invoked when

Further Readings

1. <http://www.javascriptkit.com>
2. <https://developer.mozilla.org/en/docs/JavaScript>
3. <http://www.dynamicdrive.com>

Reference Books

1. **JavaScript: The Complete Reference.** By Thomas Powell and Fritz Schneider, Third edition, TMH.
2. JavaScript for Programmers, “**Deitel guide to XHTML®, CSS and JavaScript™**”
3. Programming JavaScript Applications Robust Web Architecture With Node, HTML5,

and Modern JS Libraries By Eric Elliott Publisher: O'Reilly Media.

4. HTML5 Programming with JavaScript For Dummies By John Paul Mueller
Publisher: Wiley / For Dummies.
5. Training Guide: Programming in HTML5 with JavaScript and CSS3 By Glenn Johnson
Publisher: Microsoft Press Released: January 2013.

Additional Online Reading Resources

1. <http://en.wikipedia.org/wiki/JavaScript>
2. <http://www.w3schools.com/js>
3. <http://www.javascriptkit.com>
4. <https://developer.mozilla.org/en/docs/JavaScript>
5. <http://www.dynamicdrive.com>

Test Your Skills Exercise: Programming

Q. 1: Define the characteristics of JavaScript? Is java and javascript both are similar? Write the difference between both of them.

Q. 2: Point the errors in the following code and write the corrected script:

```
<SCRIPT LANG="JavaScript">
var A=10;
var B=6;
switch A%B
{
case 0;
document.write("The input is:");
break;
case 1;
document.write("The color is:")
break;
else;
document.write("You didn't choose")
}
```

```
</SCRIPT>
```

Q. 3: Give the output of the following code segment:

```
<BODY>
```

```
<SCRIPT LANGUAGE="JavaScript">
```

```
var sum = new Array(6);
```

```
var Total = 0;
```

```
sum[0] = 0;
```

```
for(var icount = 1; icount < 4; icount++)
```

```
{
```

```
    Total += icount;
```

```
    sum[icount] = Total;
```

```
    document.write(sum[icount]+"<br>");
```

```
}
```

```
document.write(Total);
```

```
</SCRIPT>
```

```
</BODY>
```

Q. 4: (a) Explain the functions and arrays in javascript with example?

(b) How the forms are handled in javascript? Explain.

Q. 5: How can javascript make a website easier to use? That is, are there certain javascript techniques that make it easier for people to use a website?

Q. 6: How do you convert numbers between different bases in javascript?

Q. 7: How do we get javascript on a web page explain?

Q. 8: Write a program in javascript using various data types to find the round off integer values?

Q. 9: Write the program in javascript to solve the given polynomial:

Q. 10: Design a home page of your school. Include some images/pictures and other events of an academic calendar for session 2013-2014. Also include some audio/video features to make your web-site more interactive.

4

WORK INTEGRATED LEARNING

IT – WA-I

Learning objectives

After studying this unit the students will be able to achieve the following objectives:

- ◆ To understand the various work areas where the students can fit themselves after having working knowledge of software development tools namely; multimedia authoring tools, html, CSS and javascript.
- ◆ To understand various domains in which students can use their acquired programming skills for the development of different software applications specially web applications to cater the needs of the Business
- ◆ To explore various projects in different domains which can be developed by the students using multimedia authoring tools, html, CSS and javascript.

Introduction

- ◆ Now a days, there are enormous job/employment opportunities in the market to develop various web applications catering the business needs. In order to develop web application, the student must be exposed to web applications development tools such as multimedia authoring tools, html, CSS and javascript. There are different domains such as human resources, Education, Banking, Insurance, Retail Chain, Planning, Finance, Media, Marketing, Advertising, Medical etc. in which students can use their acquired programming and functional skills of multimedia authoring tools, html, CSS and javascript for development of web applications.

Resource requirement

Hardware: Computer system with latest/sufficient configuration supported by power backup

Relevant Softwares: Operating System(windows/linux), Java development tool kit, multimedia authoring tools, web browsers like Internet Explorer, Mozilla, Google Chrome etc.

E1. Identification of Work Areas

Suitable web applications can be developed for improving efficiency and effectiveness of the organizations working in different domains. The purpose of developing software(web applications) is to maintain huge information related to the business domain, infrastructure, manpower etc. The software system/application must provide useful information for discharging various functions in the organization for achieving business/social goals. The different work areas/domains are identified with typical requirements as mentioned below, which are indicative and not exhaustive in nature.

1. Human resources domain, which may involve maintaining information related to unskilled labor, skilled labor, supervisory and middle level of work force, professionals etc. along with various kind of industries where persons are employed.
2. Education domain, which may involve elementary, primary, undergraduate, post graduate and doctoral level. The focus should be to maintain the information related to teachers, students and schools, colleges at the level of city, state or country level. The information may be needed to extract from regulatory authorities like CBSE, MHRD, AICTE, UGC etc.
3. Banking Domain, which may involve maintaining information related to various kinds of customers, various schemes which are offered by the Banking organization. Bank can take different kind of decisions on the basis of information related to customers and schemes for example, which scheme is attractive?
4. Insurance Domain, which may involve maintaining information related to various kinds of products offered(policies) with benefits and other terms and conditions.
5. Hospitals Domain, which may involve maintaining information related to various kinds of patients, doctors, specializations etc.
6. Retail Chain Business maintains information related to various outlets in the city, employee, product etc.

E2. Work Experience

The students can have working exposure in the above mentioned work areas, where they would get an opportunity to develop various web applications. The list of web

applications, which can be developed by the students are mentioned below. The following list is indicative not exhaustive:

1. Development of web based human resources management system.
2. Development of web based education facilitation system
3. Development of web based bank information system.
4. Development of web based insurance information system (web portal)
5. Development of web based retail chain management system
6. Development of web based hospital information system
7. Development of web based hotel reservation system
8. Development of web based railway ticket reservation system
9. Development of web based Application for managing advertising activities
10. Development of web based application for managing placement activities

5

CASE STUDY: DESIGNING A WEB-SITE FOR YOUR SCHOOL

One of the most important steps involved in creating a school web site is deciding what content will be included and how it will be organized. Building a school web site provides a unique addition to the curriculum. In the development phase, students and teachers should discuss the kinds of materials they want to add to the site. Students may consider the following issues while designing web-site for your own school.

- ◆ **School information:** General information such as the school's telephone number and address along with driving directions should be included. Mentioning of the school boundaries is also useful. The school's history, mission statement, and awards received are interesting and serve to promote a feeling of community. The names of the administration and faculty along with their school email addresses may also be added. Other information about the office hours, the bell and bus schedules, and after-school care may also be included.
- ◆ **School policy:** The school policy is a very important document that may be made readily available by posting it on the school site. School policy information might include use of the Web in school, admission procedures/ requirements, dress code, absences, and behavior expectations. Publishing school policies on your site can help increase awareness and access to this information.
- ◆ **Calendar of events:** A calendar of events provides parents, teachers and students with an easy way to keep up-to-date with the happenings at the school. Special events such as school plays, sports, field trips, and standardized tests are just a few examples of what may be included. The school lunch menu may be added to the calendar as well. Calendars are easily updated each month and this method of showing events cuts down on the amount of paper and printing that needs to be done. In addition, parents can always have access to the most recent calendar without worrying about it becoming lost.

- ◆ **Extracurricular activities:** School clubs or organizations involved in organizing special events may create their own pages describing their purposes and anticipated activities for the entire year. Meeting and event schedules with detailed information are useful for informing students and gaining their interest.
- ◆ **Newsletters:** Newsletters written by the students are another great way to relay information about the school. Students may submit articles, reports on class trips, and special school events. Involving students gives them a chance to share their thoughts in writing and build school spirit. A message from the principal that provides motivation and encouragement to students and promotes parent involvement may be included.
- ◆ **Parent involvement:** An excellent way to encourage parent involvement is to keep them informed of opportunities such as volunteering, PTA meetings, and fundraising activities. By providing a wide range of choices, the school can help parents find the activities that fit their time and schedule constraints as well as their interests.
- ◆ **Links:** There are many excellent resources available on the Web for teachers, students and parents. Providing links to some of these sources is very helpful. Be sure to check the links periodically to make sure the resources are still available. Teachers can have easy access to learning communities and lesson plan ideas. Students can link to reference or ask-an-expert sites and sites with virtual museums. Links of interest to parents can also be included.





CENTRAL BOARD OF SECONDARY EDUCATION

Shiksha Kendra, 2, Community Centre, Preet Vihar, Delhi-110301

Tel.: 91-11-22509252-59 Fax : 91-11-22515826

E-mail : [cbse- @nda.vsnl.net.in](mailto:cbse@nda.vsnl.net.in) website : www.cbse.nic.in