

# COMPLETE JAVASCRIPT

14 HOURS



MYNTRA  
PROJECT

CERTIFICATE



NOTES



[Video Link](#)

# KG Coding

Some Other One shot Video Links:

- [Complete HTML](#)
- [Complete CSS](#)
- [Complete JavaScript](#)
- [Complete React and Redux](#)
- [One shot University Exam Series](#)

<http://www.kgcoding.in/>

Our  YouTube Channels

[KG Coding Android App](#)



[KG Coding](#)



[Knowledge GATE](#)



[KG Placement Prep](#)

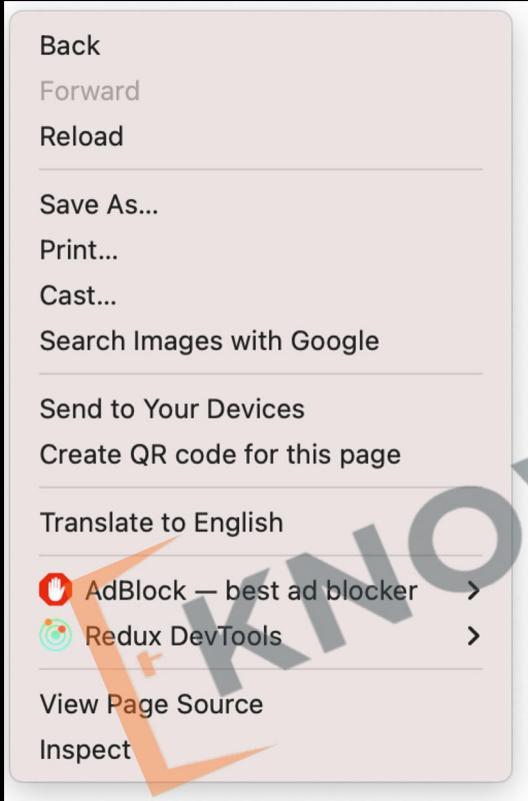


[Sanchit Socket](#)

# Introduction to JavaScript

1. JS in Console
2. DOM Manipulation
3. Chrome Extensions
4. What is a Programming Language?
5. What is Syntax?
6. HTML/CSS/JavaScript

# 1. JS in Console (Inspect)



1. Allows real-time editing of HTML/CSS/JS
2. Run Scripts: Test code in console.
3. Debug: Locate and fix errors.
4. Modify DOM: Change webpage elements.  
Errors: View error messages.

# 1. JS in Console (Alert)

A screenshot of a Google search results page. The URL in the address bar is "google.com". The main content shows the Google logo and a search bar with a microphone and camera icon. Below the search bar are buttons for "Google Search" and "I'm Feeling Lucky". At the bottom, there's a multilingual offer: "Google offered in: हिन्दी वांला ತೆಲುಗು ಮರಾಠಿ ತமிழ் ગુજરાતી ਕਨ੍ਨಡ ಮಲಯಾളം ਪੰਜਾਬੀ". On the right side of the screen, the browser's developer tools are open, specifically the "Console" tab. An alert dialog box is displayed in the center, reading "www.google.com says JavaScript is Magical" with an "OK" button. The console area shows the command `> alert('JavaScript is Magical');` and its result `>`.

```
> alert('JavaScript is Magical');
```

OK

www.google.com says  
JavaScript is Magical

Gmail Images

Default levels ▾ | No Issues

OK

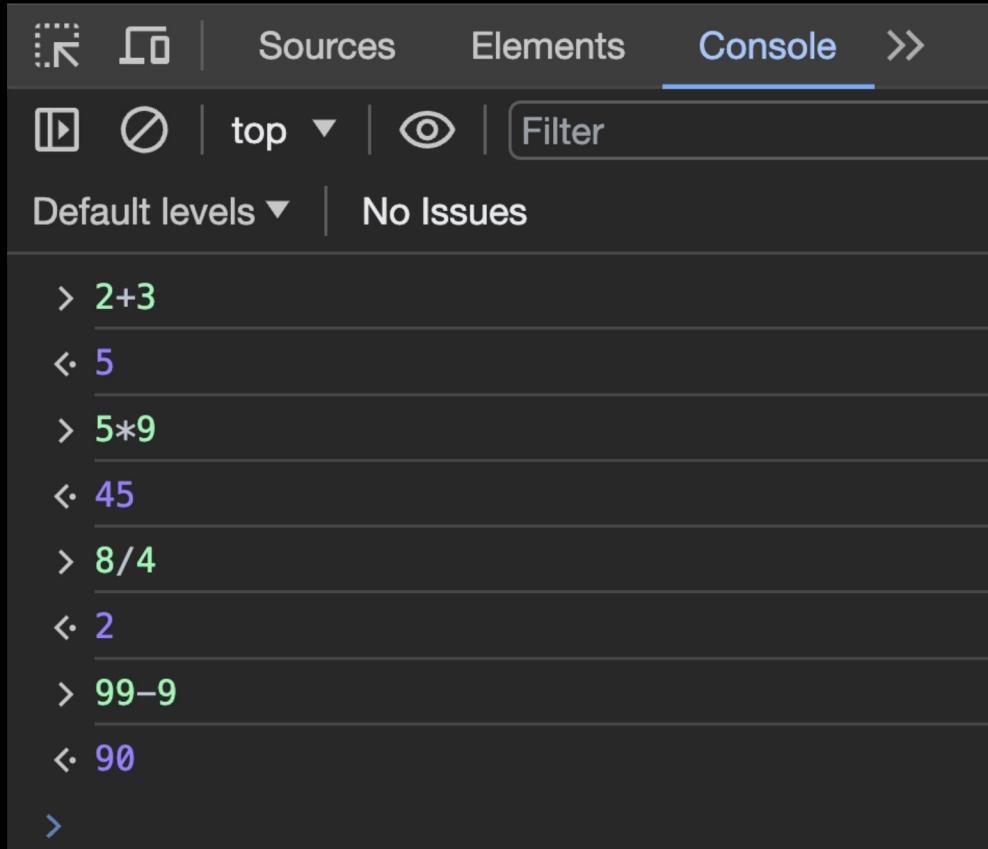
Filter

sources Console >

Google Search I'm Feeling Lucky

Google offered in: हिन्दी वांला ತೆಲುಗು ಮರಾಠಿ ತಮಿಳು ગુજરાતી ਕਨ੍ਨಡ ಮಲയാളം ਪੰਜਾਬੀ

# 1. JS in Console (Math)

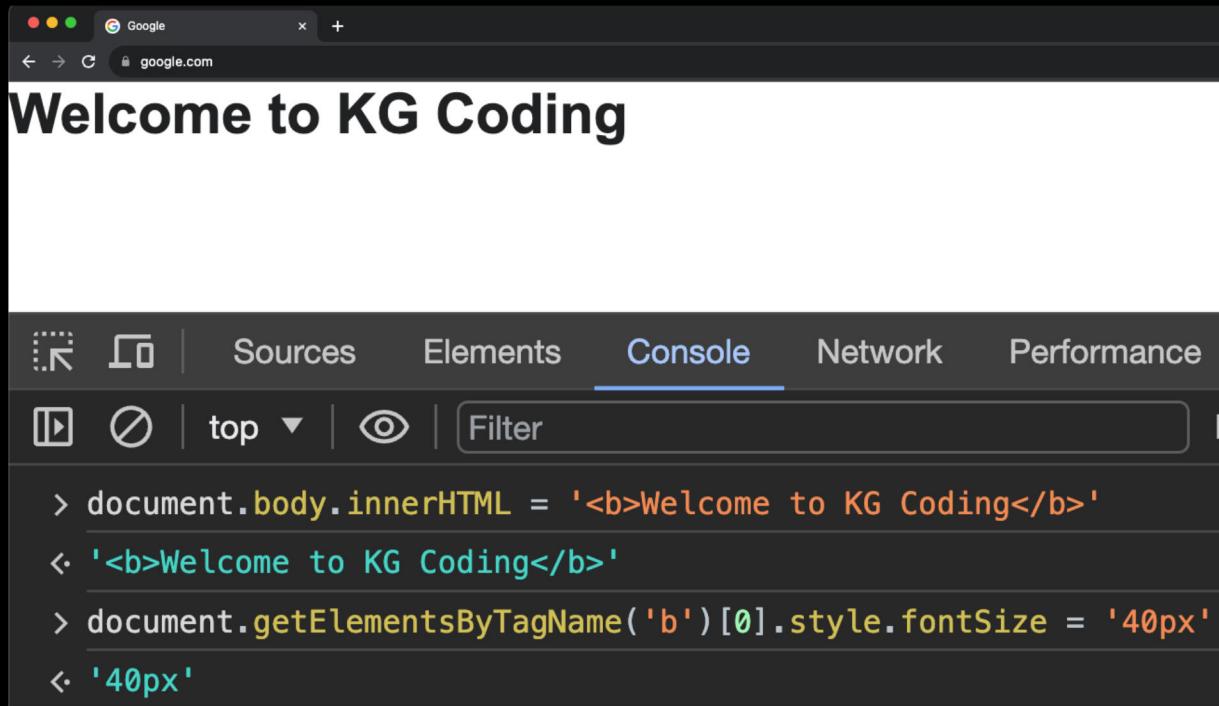


A screenshot of a browser's developer tools interface, specifically the 'Console' tab. The console window has a dark background with light-colored text. At the top, there are icons for refresh, zoom, and navigation, followed by tabs for 'Sources', 'Elements', 'Console' (which is underlined in blue), and '>>'. Below the tabs are more icons: a play button, a stop button, a dropdown menu set to 'top', an eye icon, and a 'Filter' input field. Underneath these are two status indicators: 'Default levels ▾' and 'No Issues'. The main area contains a list of previous commands and their results:

- >  $2+3$   
↳ 5
- >  $5*9$   
↳ 45
- >  $8/4$   
↳ 2
- >  $99-9$   
↳ 90
- >

Console can be used  
as a Calculator

# 2. DOM Manipulation

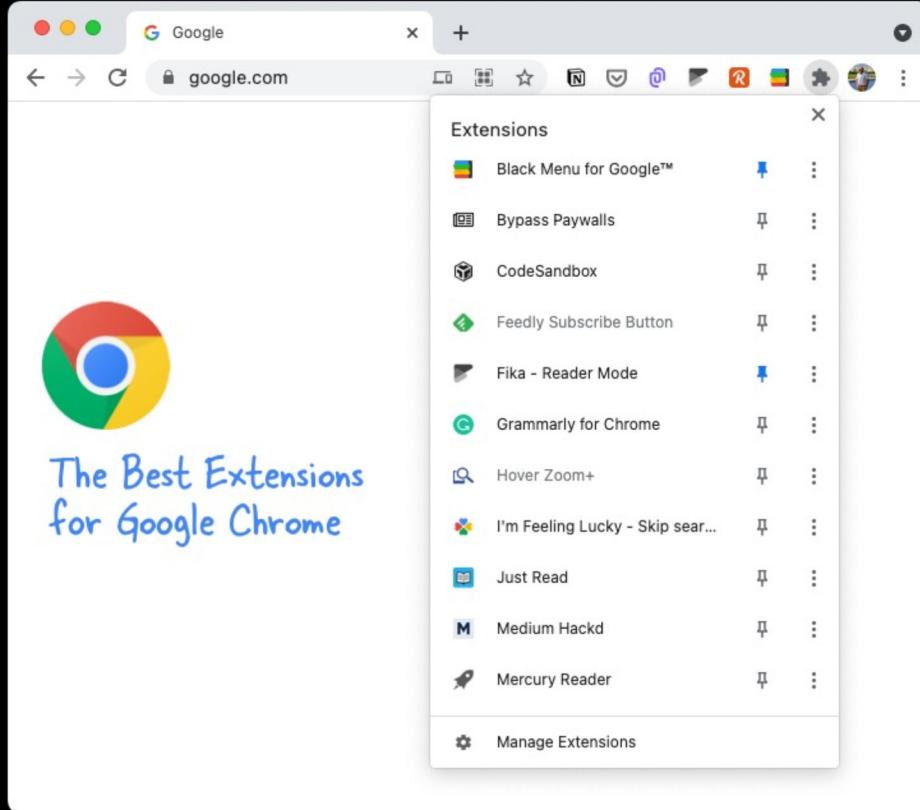


A screenshot of a browser window showing the developer tools console tab selected. The page content is "Welcome to KG Coding". The console output shows two lines of JavaScript code: changing the innerHTML of the body to bold text and setting the font size of the first  **element to 40px.**

```
> document.body.innerHTML = '<b>Welcome to KG Coding</b>'  
< ' <b>Welcome to KG Coding</b>'  
> document.getElementsByTagName('b')[0].style.fontSize = '40px'  
< '40px'
```

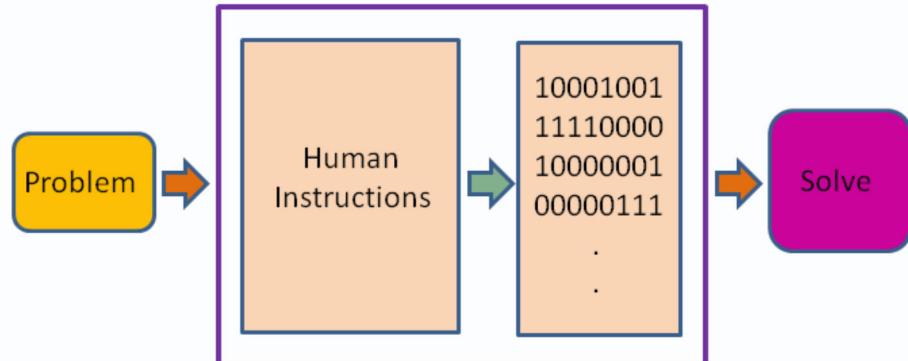
1. Change **HTML**
2. Change **CSS**
3. Perform **Actions**

# 3. Chrome Extensions



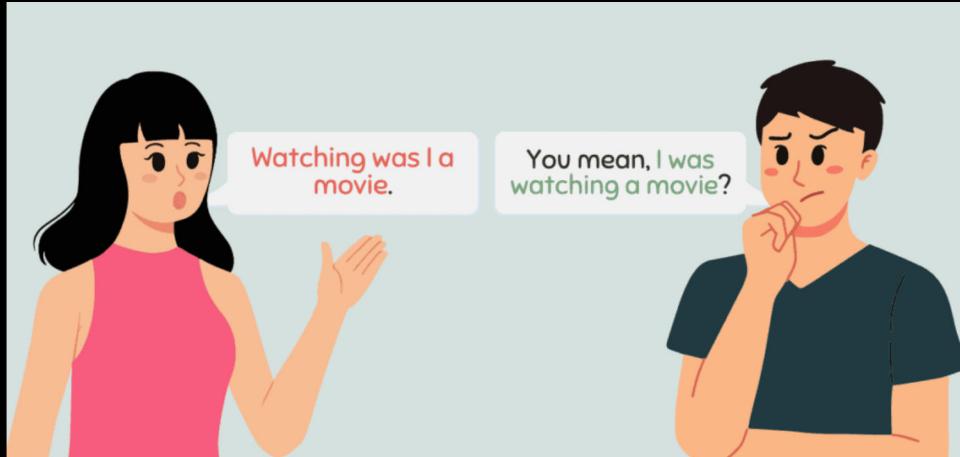
1. Create Features: Add new functionalities to Chrome.
2. Interact with Web: Modify or read webpage content.
3. API Access: Use Chrome's built-in functions.
4. User Experience: Enhance or customize browsing.

# 4. What is a Programming Language?



- Giving instructions to a computer
- **Instructions:** Tells computer what to do.
- These instructions are called **code**.

# 5. What is a Syntax?



- Structure of words in a sentence.
- Rules of the language.
- For programming exact syntax must be followed.

```
> alert 'hello world'  
✖ Uncaught SyntaxError: Unexpected string  
> |
```

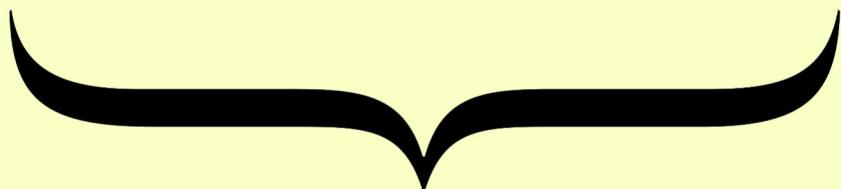
# 6. FrontEnd / BackEnd / FullStack



**Client Side / Front-End  
Web Development**



**Server Side  
Back-End**



**Full Stack**

# 6. Role of Browser



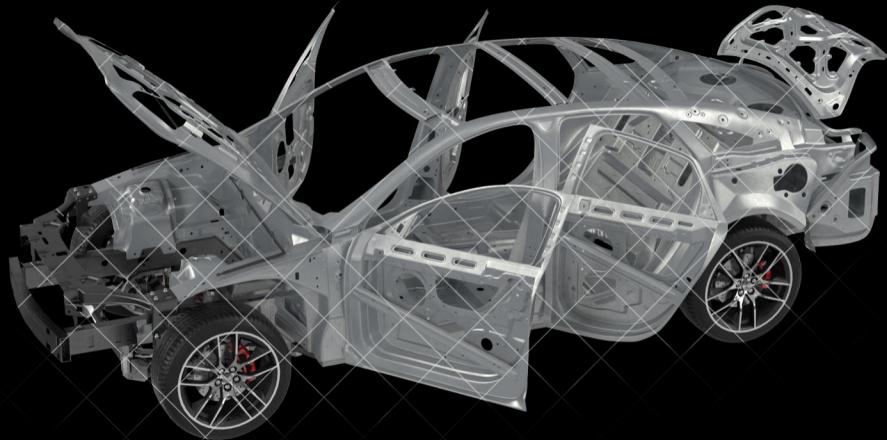
1. **Displays Web Page:** Turns HTML code into what you see on screen.
2. **User Clicks:** Helps you interact with the web page.
3. **Updates Content:** Allows changes to the page using JavaScript.
4. **Loads Files:** Gets HTML, images, etc., from the server.



# 6. HTML

(Hypertext Markup Language)

1. **Structure:** Sets up the layout.
2. **Content:** Adds text, images, links.
3. **Tags:** Uses elements like `<p>`, `<a>`.
4. **Hierarchy:** Organizes elements in a tree.



**HTML** is required for **JavaScript**

**COMPLETE**

**5 HTML**

**4 HOUR**

**PROJECT**

**CERTIFICATE**

**CODE**



**NOTES**

**React.Frag**

**Ex- amazon Microsoft**





# 6. CSS

(Cascading Style Sheets)

1. **Style:** Sets the look and feel.
2. **Colors & Fonts:** Customizes text and background.
3. **Layout:** Controls position and size.
4. **Selectors:** Targets specific **HTML** elements.



**CSS** is required for **JavaScript**

# COMPLETE CSS 7 HOUR

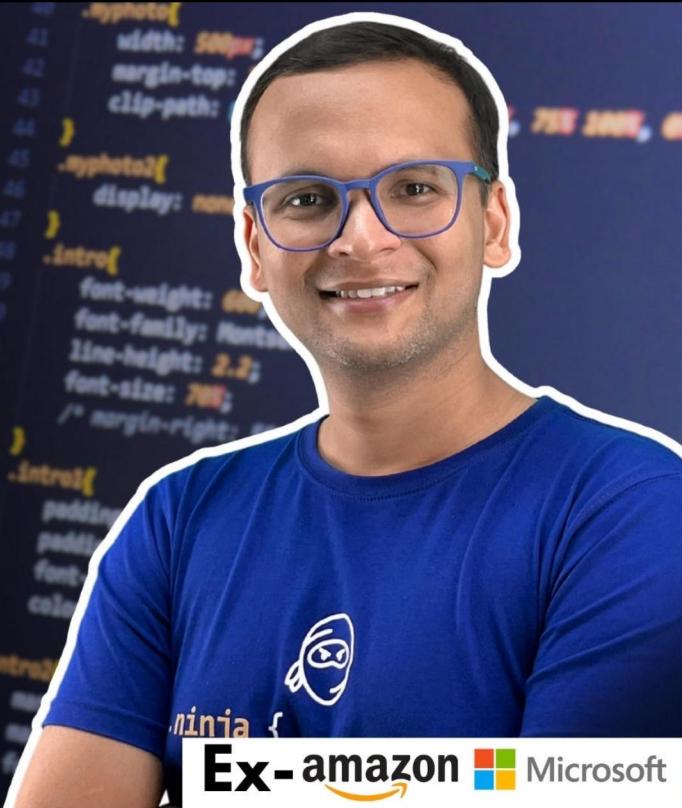


MYNTRA  
PROJECT



CERTIFICATE

NOTES



Ex- amazon Microsoft



# 6. JS

(Java Script)

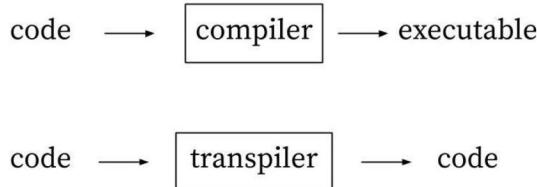
1. JavaScript has nothing to do with Java
2. Actions: Enables interactivity.
3. Updates: Alters page without reloading.
4. Events: Responds to user actions.
5. Data: Fetches and sends info to server.



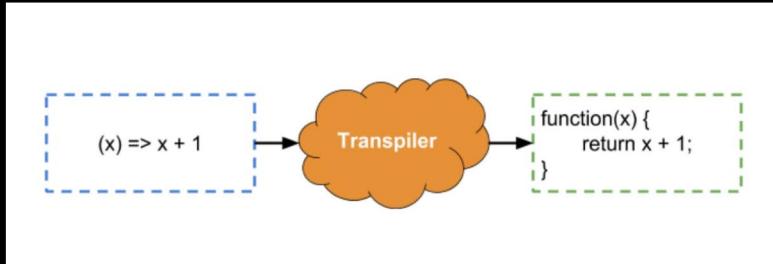


# 6. JS

## (Java Script)



1. JavaScript runs at the **client side** in the browser.
2. Coffee Script / **TypeScript** are **transpiled** to JavaScript.



# Introduction Revision

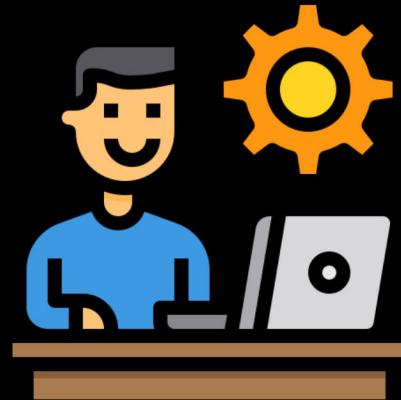
1. JS in Console
2. DOM Manipulation
3. Chrome Extensions
4. What is a Programming Language?
5. What is Syntax?
6. HTML/CSS/JavaScript



# Practice Exercise

## Introduction

1. Use an **alert** to display Good Morning.
2. **Display** your name in a **popup**.
3. Using Math calculate the following:  
=> 75-25  
=> 3+3-5
4. Change **Facebook** page to display “I am Learning JS”



# KG Coding

Some Other One shot Video Links:

- [Complete HTML](#)
- [Complete CSS](#)
- [Complete JavaScript](#)
- [Complete React and Redux](#)
- [One shot University Exam Series](#)



<http://www.kgcoding.in/>

Our YouTube Channels

[KG Coding Android App](#)



[KG Coding](#)



[Knowledge GATE](#)



[KG Placement Prep](#)



[Sanchit Socket](#)

# Numbers & Strings

7. Arithmetic Operators
8. Order of Operations
9. Types of Numbers (Integers & Floats)
10. Don't learn syntax.
11. Strings
12. TypeOf Operator

# 7. Arithmetic Operators

Operators	Meaning	Example	Result
+	Addition	4+2	6
-	Subtraction	4-2	2
*	Multiplication	4*2	8
/	Division	4/2	2
%	Modulus operator to get remainder in integer division	5%2	1



✓ 2/2 ITEMS SELECTED

REMOVE

MOVE TO WISHLIST



Allen Solly

Men Textured Slim Fit Mid-Rise Trousers

Sold by: Westbury Holdings Pvt Ltd\_Madura

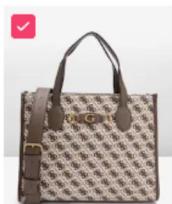
Size: 32 - Qty: 1 -

₹1,474 ₹2,499 41% OFF

Coupon Discount: ₹29

⌚ 14 days return available

✓ Delivery by 8 Oct 2023



GUESS

Brand Logo Woven Design Structured Handheld Bag

Sold by: Supercom Net

Size: Onesize - Qty: 1 - 9 left

₹14,039 ₹15,599 10% OFF

Coupon Discount: ₹272

⌚ 7 days return available

🚚 Delivery by 3 Oct 2023

#### COUPONS



1 Coupon applied

You saved additional ₹301

EDIT

#### GIFTING & PERSONALISATION



Yay! Gift Wrapping applied

Your order will be gift wrapped with your personalised message

REMOVE

EDIT MESSAGE

#### PRICE DETAILS (2 Items)

Total MRP ₹18,098

Discount on MRP -₹2,585

Coupon Discount ⓘ -₹301

Gift Wrap Charges ₹25

Convenience Fee [Know More](#) ₹20

**Total Amount ₹15,257**

PLACE ORDER

# 8. Order of Operations

B	O	D	M	A	S
Bracket	Order	Divide	Multiply	Add	Subtract
( )	$\sqrt{x}$ $x^2$	$\div$ or $\times$	$+$ or $-$		
Parentheses	Exponents	Multiply	Divide	Add	Subtract
P	E	M	D	A	S

$$9 \div 3 \times 2 \div 6$$



$$8 - 5 + 7 - 1$$



✓ 2/2 ITEMS SELECTED

REMOVE

MOVE TO WISHLIST



Allen Solly

Men Textured Slim Fit Mid-Rise Trousers  
Sold by: Westbury Holdings Pvt Ltd\_Madura

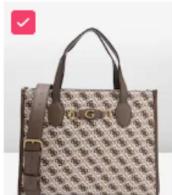
Size: 32 - Qty: 1 -

₹1,474 ₹2,499 41% OFF

Coupon Discount: ₹29

⌚ 14 days return available

✓ Delivery by 8 Oct 2023



GUESS

Brand Logo Woven Design Structured Handheld Bag  
Sold by: Supercom Net

Size: Onesize - Qty: 1 - 9 left

₹14,039 ₹15,599 10% OFF

Coupon Discount: ₹272



⌚ 7 days return available

🚚 Delivery by 3 Oct 2023

#### COUPONS



1 Coupon applied

You saved additional ₹301

EDIT

#### GIFTING & PERSONALISATION



Yay! Gift Wrapping applied

Your order will be gift wrapped with your personalised message

REMOVE

EDIT MESSAGE

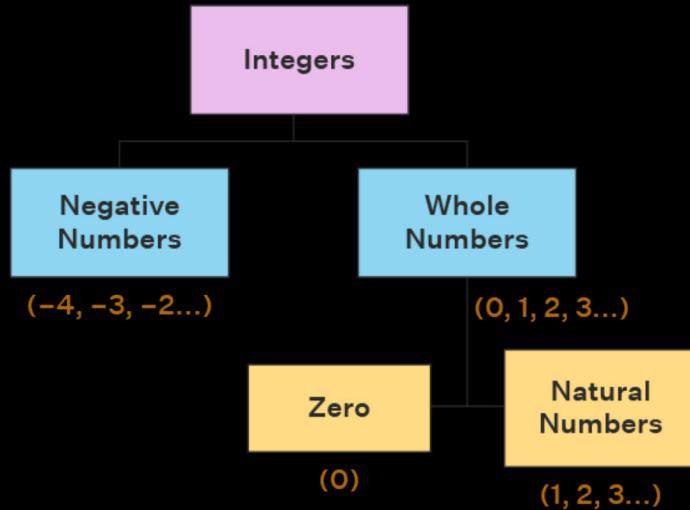
#### PRICE DETAILS (2 Items)

Total MRP	₹18,098
Discount on MRP	-₹2,585
Coupon Discount ⓘ	-₹301
Gift Wrap Charges	₹25
Convenience Fee <a href="#">Know More</a>	₹20

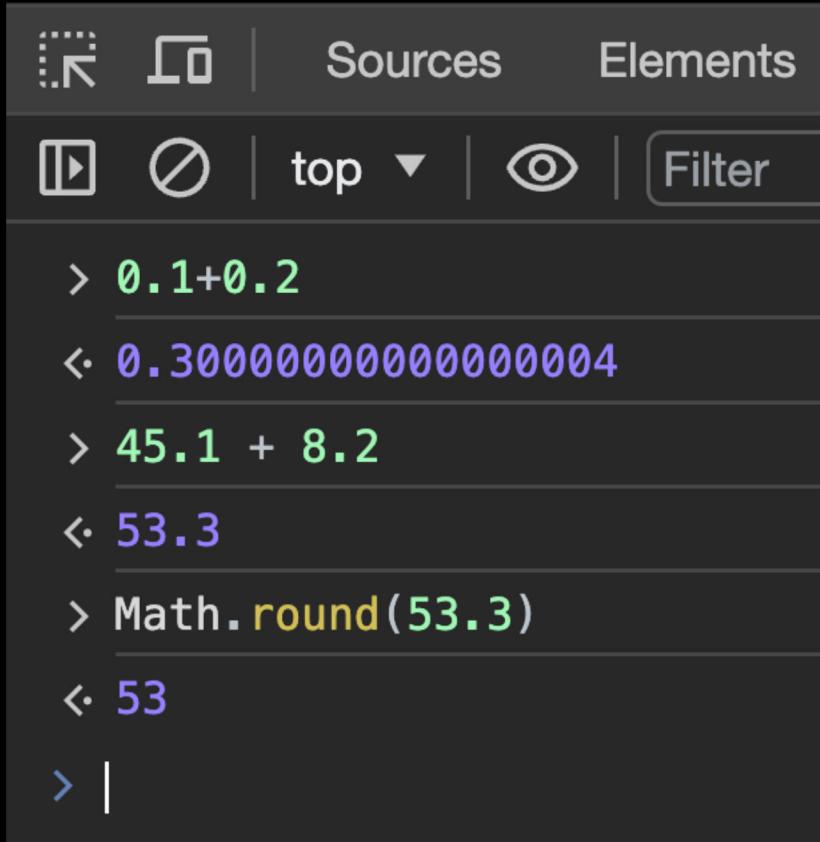
Total Amount ₹15,257

PLACE ORDER

# 9. Types of Numbers (Integers & Floats)



# 9. Types of Numbers (Float Problems)

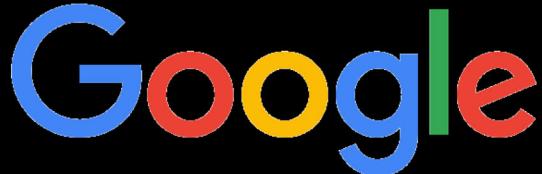


A screenshot of a browser's developer tools console. At the top, there are icons for file operations (New, Open, Save, Close) followed by 'Sources' and 'Elements' tabs. Below that is a toolbar with icons for back, forward, stop, refresh, and a 'Filter' button. The main area shows the following command history:

- > `0.1+0.2`
- < `0.3000000000000004`
- > `45.1 + 8.2`
- < `53.3`
- > `Math.round(53.3)`
- < `53`
- > |

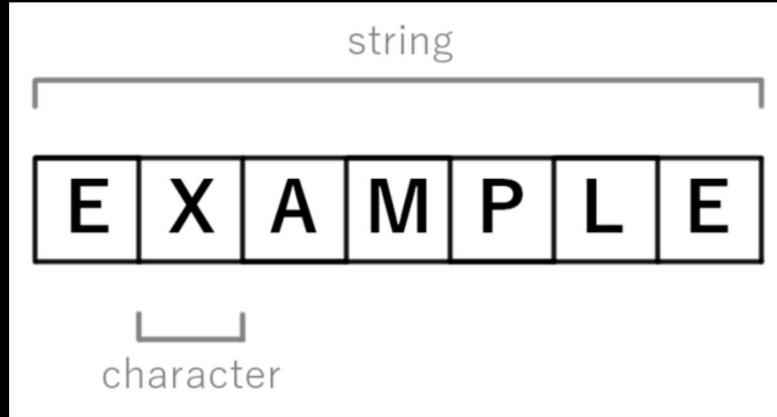
1. JavaScript has many problems with float operations.
2. We can use `Math.round()` to convert floats to integers.
3. Always do money calculation in Paisa instead of Rupees

# 10. Don't learn syntax.



1. **Google**: Quick answers to coding problems.
2. **MDN**: In-depth guides and documentation.  
<https://developer.mozilla.org/>
3. **ChatGPT**: Real-time assistance for coding queries.
4. **Focus**: Understand concepts, not just syntax.

# 11. Strings



1. Strings hold **textual data**, anything from a single character to paragraph.
2. Strings can be defined using **single quotes ''**, **double quotes " "**, or **backticks ``**. Backticks allow for template literals, which can include variables.
3. You can combine (**concatenate**) strings using the **+** operator. For example, **"Hello" + " World"** will produce **"Hello World"**.

# 12. Type Of Operator

JS

## typeof function

typeof 'this is a string'

'string'

typeof 12345

'number'

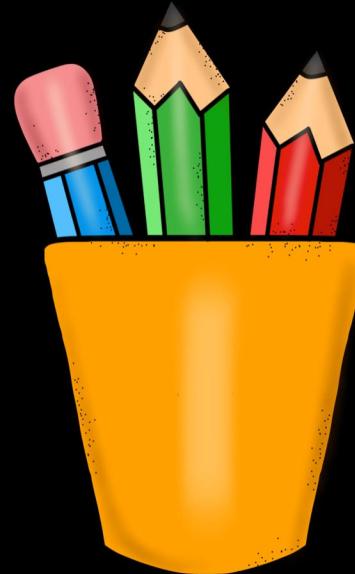
typeof (11 === 11)

'boolean'

1. **Check Type:** Tells you the data type of a variable.
2. **Syntax:** Use it like `typeof` variable.
3. **Common Types:** Returns "`number`," "`string`," "`boolean`," etc.

# Numbers & Strings Revision

7. Arithmetic Operators
8. Order of Operations
9. Types of Numbers
10. Don't learn syntax.
11. Strings
12. TypeOf Operator



# Practice Exercise

## Numbers & Strings

1. At a restaurant you ate: 1 Dal ₹ 100, 2 Roti ₹10 each, 1 Ice Cream ₹30, calculate and display the final bill amount.
2. Calculate 18% GST on iPhone15 ₹79,990 and 2 Air pods Pro ₹24990 each.
3. Create strings using all 3 methods.
4. Concatenate String with Strings, and String with numbers.
5. Create Order Summary String for our Myntra Cart.
6. Display order summary in a popup



# KG Coding

Some Other One shot Video Links:

- [Complete HTML](#)
- [Complete CSS](#)
- [Complete JavaScript](#)
- [Complete React and Redux](#)
- [One shot University Exam Series](#)



<http://www.kgcoding.in/>

Our Channels

KG Coding Android App



[KG Coding](#)



[Knowledge GATE](#)



[KG Placement Prep](#)



[Sanchit Socket](#)

# JavaScript with HTML & CSS

13. VS Code IDE

14. HTML Tags Introduction

15. CSS Introduction

16. Query Selector

17. Script Tag

18. Comments

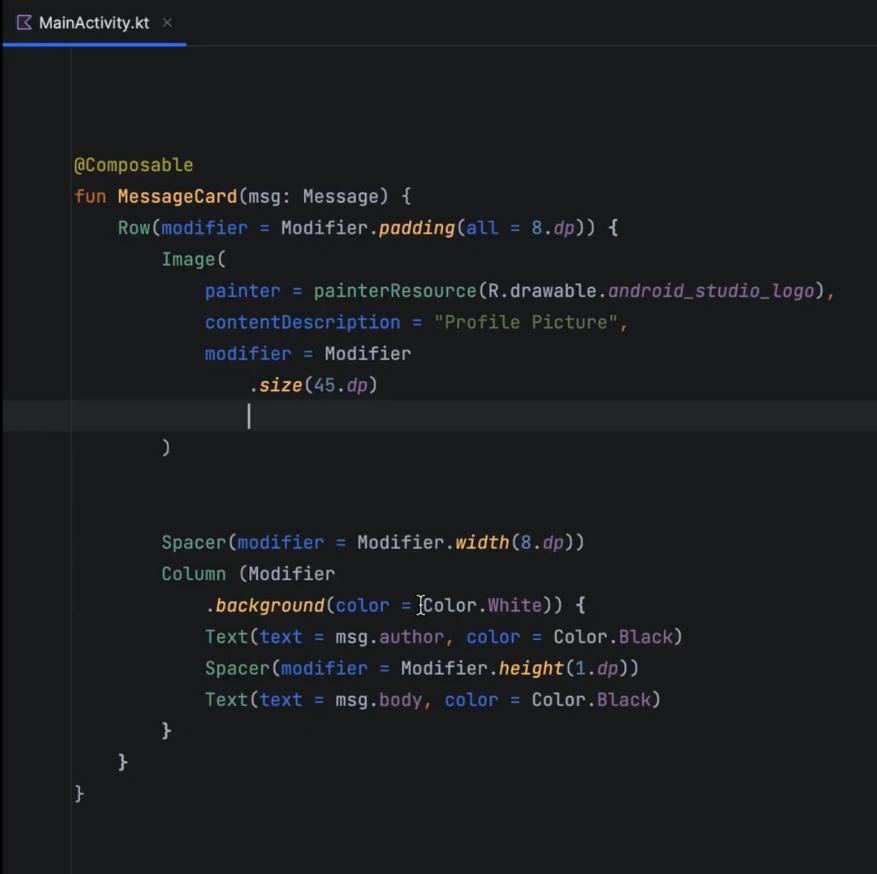
# 13. What is IDE

1. IDE stands for Integrated Development Environment.
2. Software suite that consolidates basic tools required for software development.
3. Central hub for coding, finding problems, and testing.
4. Designed to improve developer efficiency.



# 13. Need of IDE

1. Streamlines development.
2. Increases productivity.
3. Simplifies complex tasks.
4. Offers a unified workspace.
5. IDE Features
  1. Code Autocomplete
  2. Syntax Highlighting
  3. Version Control
  4. Error Checking



```
MainActivity.kt
```

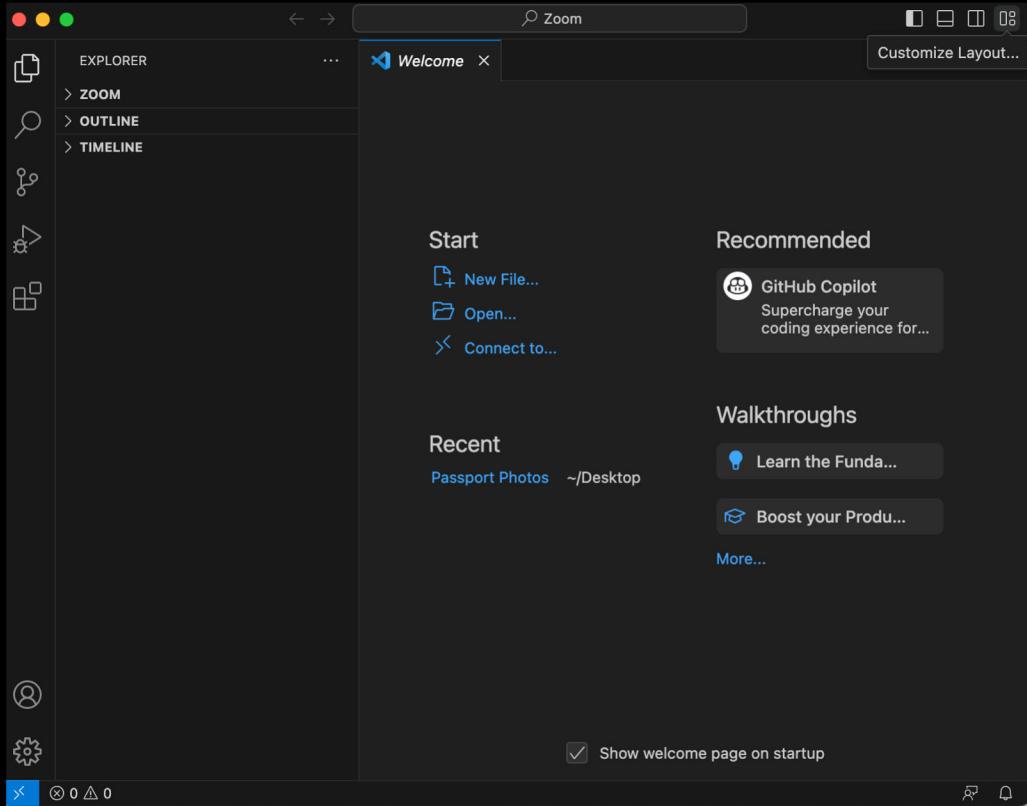
```
@Composable
fun MessageCard(msg: Message) {
    Row(modifier = Modifier.padding(all = 8.dp)) {
        Image(
            painter = painterResource(R.drawable.android_studio_logo),
            contentDescription = "Profile Picture",
            modifier = Modifier
                .size(45.dp)
        )
        Spacer(modifier = Modifier.width(8.dp))
        Column (Modifier
            .background(color = Color.White)) {
            Text(text = msg.author, color = Color.Black)
            Spacer(modifier = Modifier.height(1.dp))
            Text(text = msg.body, color = Color.Black)
        }
    }
}
```



# 13. Installation & Setup

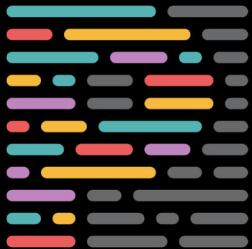
1. Search VS Code
2. Keep Your **Software** up to date

# 13. Opening project in VsCode

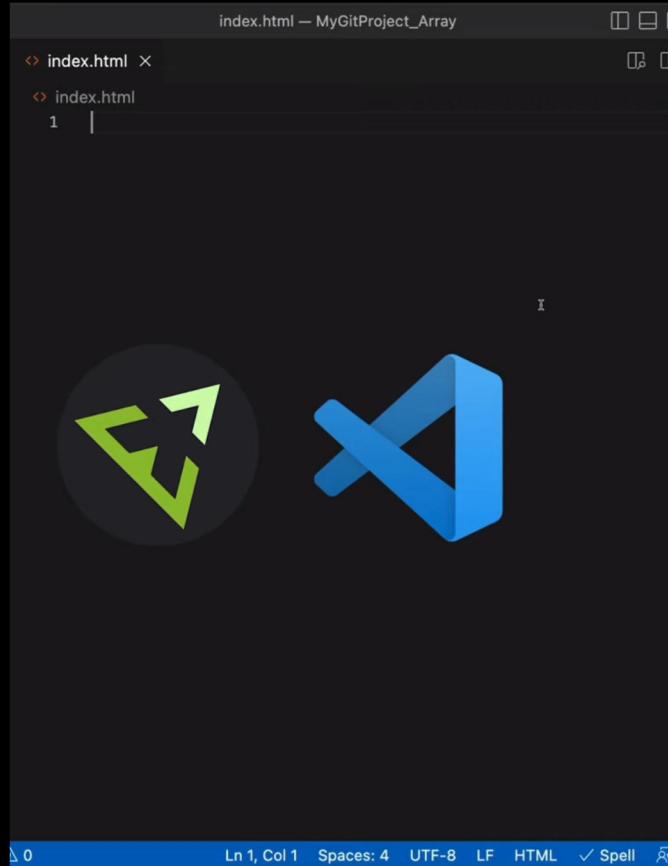


# 13. VsCode Extensions and Settings

1. Live Server
2. Prettier
3. Line Wrap
4. Tab Size from 4 to 2



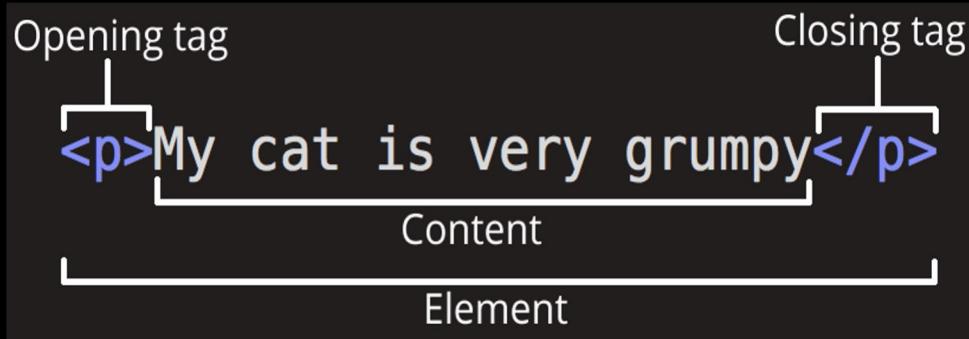
# 13. Using Emmet ! to generate code



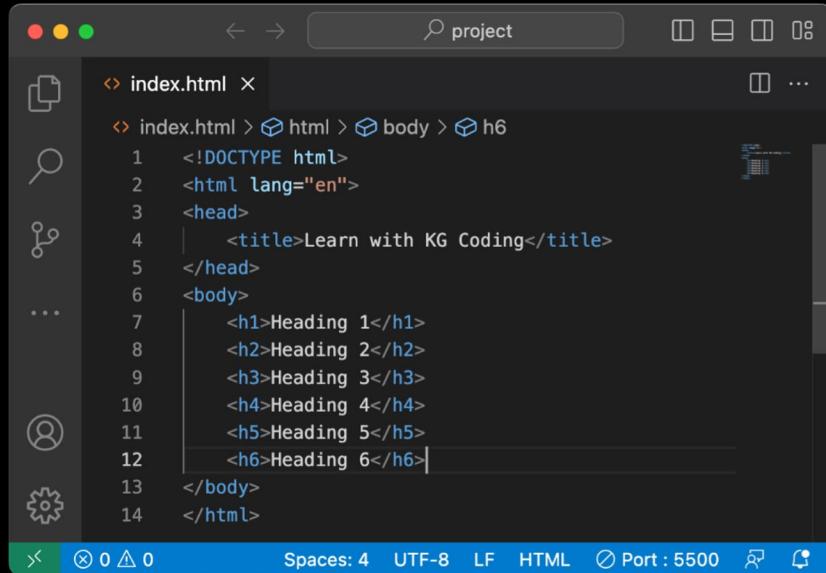
1. Type ! and wait for suggestions.

# 14 HTML Tags Introduction

1. Elements that are used to create a website are called HTML Tags.
2. Tags can contain content or other HTML tags.
3. Define elements like text, images, links



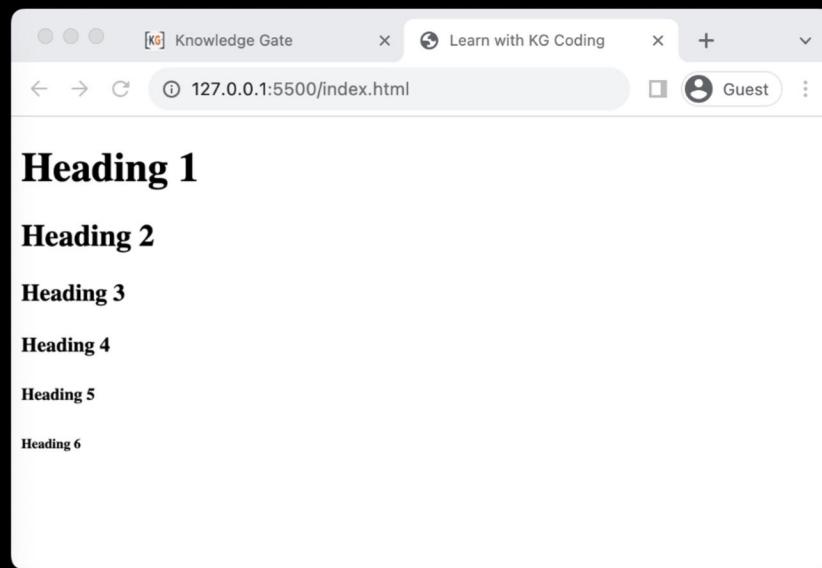
# 14 Heading Tag



A screenshot of a code editor window titled "project". The file "index.html" is open, displaying the following HTML code:

```
1 <!DOCTYPE html>
2 <html lang="en">
3   <head>
4     <title>Learn with KG Coding</title>
5   </head>
6   <body>
7     <h1>Heading 1</h1>
8     <h2>Heading 2</h2>
9     <h3>Heading 3</h3>
10    <h4>Heading 4</h4>
11    <h5>Heading 5</h5>
12    <h6>Heading 6</h6>
13  </body>
14 </html>
```

The code editor has a dark theme with light-colored syntax highlighting. The sidebar on the left contains icons for file operations like new, open, save, and search.



A screenshot of a web browser window titled "Knowledge Gate". The URL "127.0.0.1:5500/index.html" is visible in the address bar. The page displays the following rendered headings:

# Heading 1

## Heading 2

### Heading 3

#### Heading 4

##### Heading 5

###### Heading 6

1. Defines **headings** in a document
2. Ranges from **<h1>** to **<h6>**
3. **<h1>** is most important, **<h6>** is least
4. Important for **SEO**
5. Helps in structuring content

# 14 Button tag

```
Button.html — testing
Button.html •
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <title>Button</title>
5     <style>
6         button {
7             margin: 10px;
8         }
9         .blue {
10             background-color: blue;
11             color: red;
12         }
13         .wow {
14             background-color: #4CAF50;
15             border: none;
16             border-radius: 10px;
17             color: white;
18             padding: 8px 10px;
19         }
20     </style>
21 </head>
22 <body>
23     <button>Click Me</button> <br>
24     <button class="blue">Blue button</button> <br>
25     <button class="wow">Sundar button</button>
26 </body>
27 </html>
```



# 14 Paragraph Tag

The image shows a code editor on the left and a browser window on the right. The code editor displays the file 'index.html' with the following content:

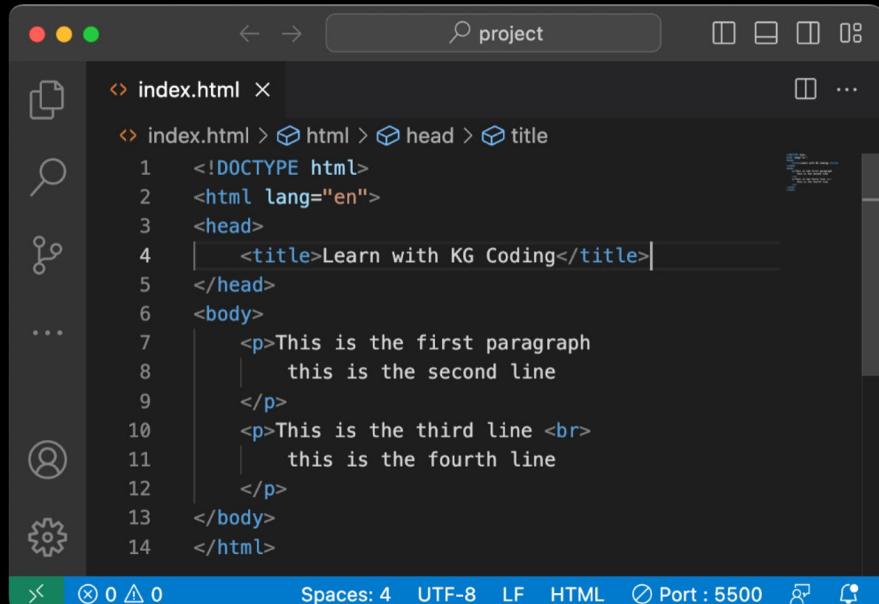
```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Learn with KG Coding</title>
</head>
<body>
    <p>This is the first paragraph</p>
    <p>This is the Second paragraph</p>
    <p>This is the Third paragraph</p>
</body>
</html>
```

The browser window shows the rendered HTML with three paragraphs:

This is the first paragraph  
This is the Second paragraph  
This is the Third paragraph

1. Used for defining **paragraphs**
2. Enclosed within **<p>** and **</p>** tags
3. Adds **automatic spacing** before and after
4. **Text wraps** to next line inside tag
5. Common in **text-heavy content**

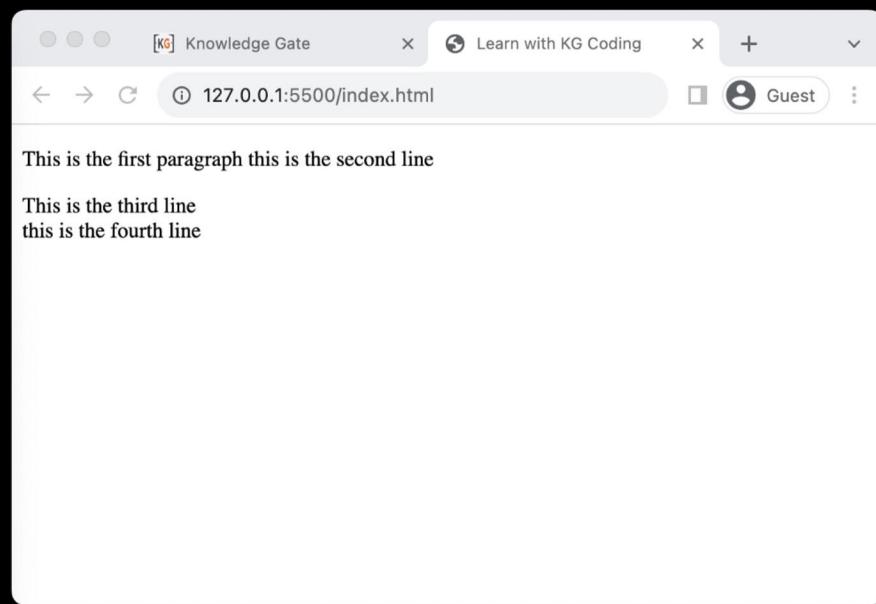
# 14 <BR> Tag



A screenshot of a code editor window titled "index.html". The code is as follows:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Learn with KG Coding</title>
</head>
<body>
    <p>This is the first paragraph
        this is the second line
    </p>
    <p>This is the third line <br>
        this is the fourth line
    </p>
</body>
</html>
```

The code editor has a dark theme with light-colored syntax highlighting. The status bar at the bottom shows "Spaces: 4", "UTF-8", "LF", "HTML", and "Port : 5500".

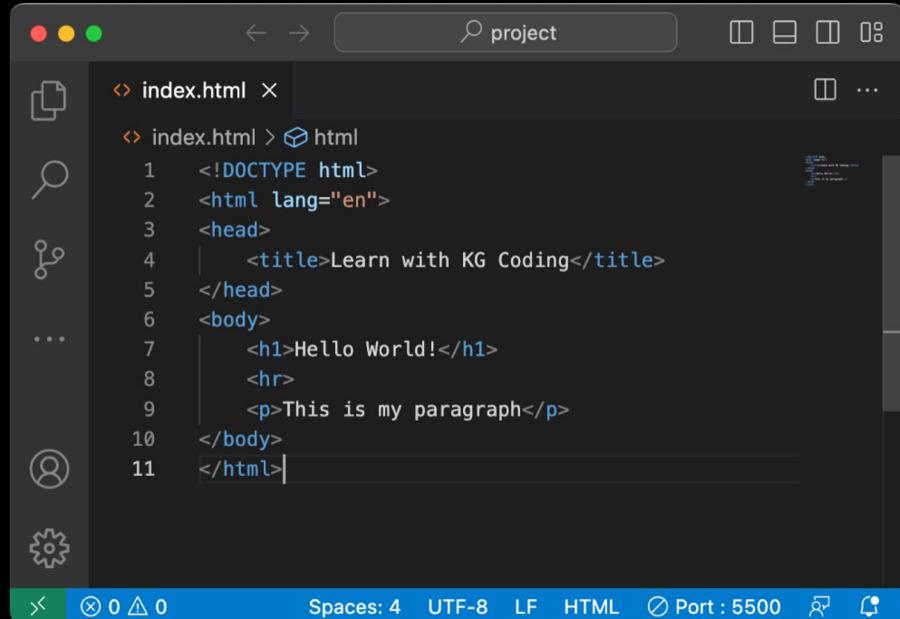


A screenshot of a web browser window titled "Knowledge Gate". The URL is "127.0.0.1:5500/index.html". The page content is:

This is the first paragraph this is the second line  
This is the third line  
this is the fourth line

1. <br> adds a line break within text
2. <br> is empty, no closing tag needed
3. <br> and <br /> are both valid

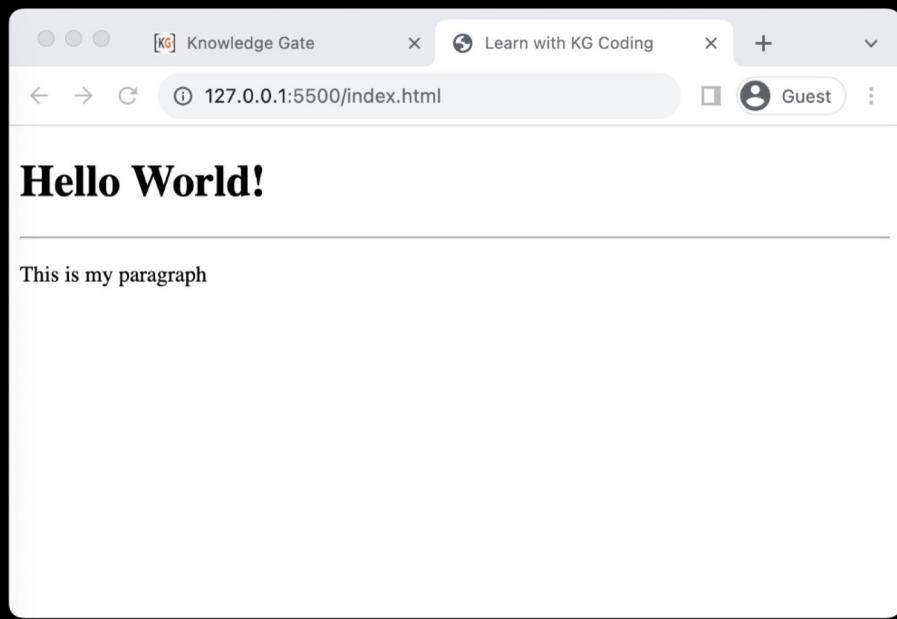
# 14 <HR> Tag



A screenshot of a code editor window titled "project". The file "index.html" is open, showing the following HTML code:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Learn with KG Coding</title>
</head>
<body>
    <h1>Hello World!</h1>
    <hr>
    <p>This is my paragraph</p>
</body>
</html>
```

The code editor has a dark theme with light-colored syntax highlighting. The status bar at the bottom shows "Spaces: 4" and "Port : 5500".



A screenshot of a web browser window titled "Knowledge Gate". The URL "127.0.0.1:5500/index.html" is visible in the address bar. The page content is:

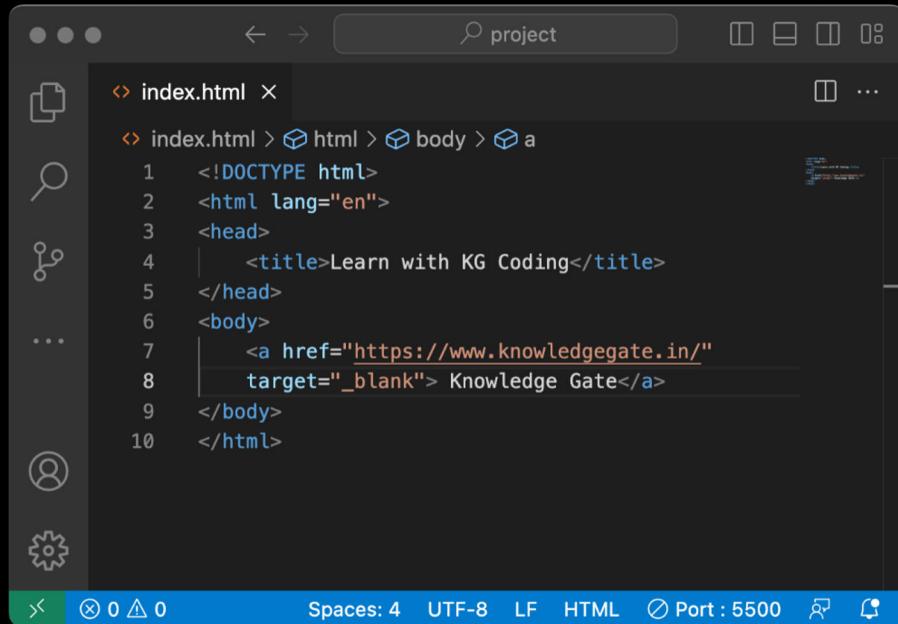
# Hello World!

---

This is my paragraph

1. <hr> creates a horizontal rule or line
2. <hr> also empty, acts as a divider

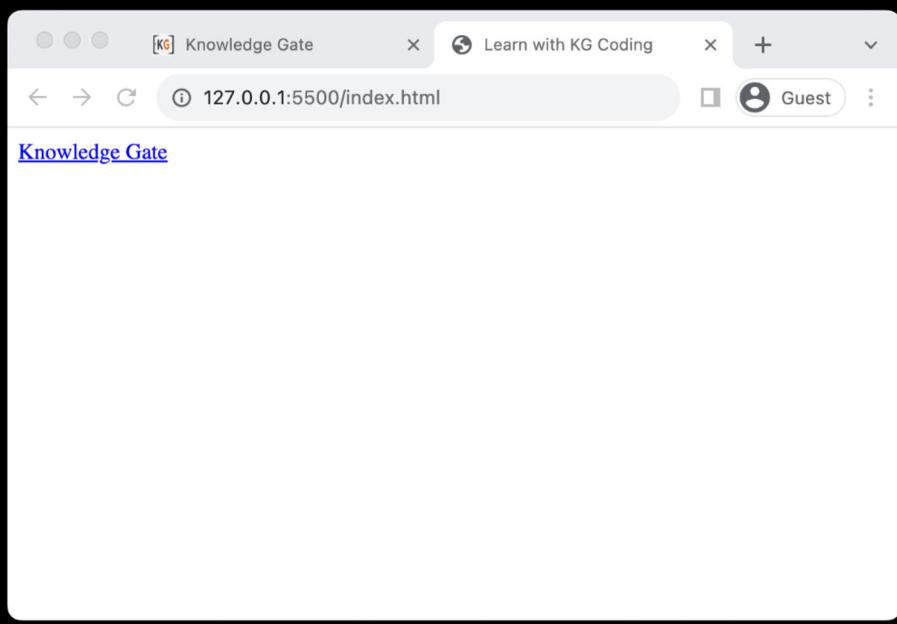
# 14 Anchor Tag



A screenshot of a code editor window titled "index.html". The code editor shows the following HTML structure:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Learn with KG Coding</title>
</head>
<body>
    <a href="https://www.knowledgegate.in/" target="_blank"> Knowledge Gate</a>
</body>
</html>
```

The code editor interface includes a sidebar with various icons for file operations, a search bar at the top, and a status bar at the bottom indicating "Spaces: 4" and "Port : 5500".



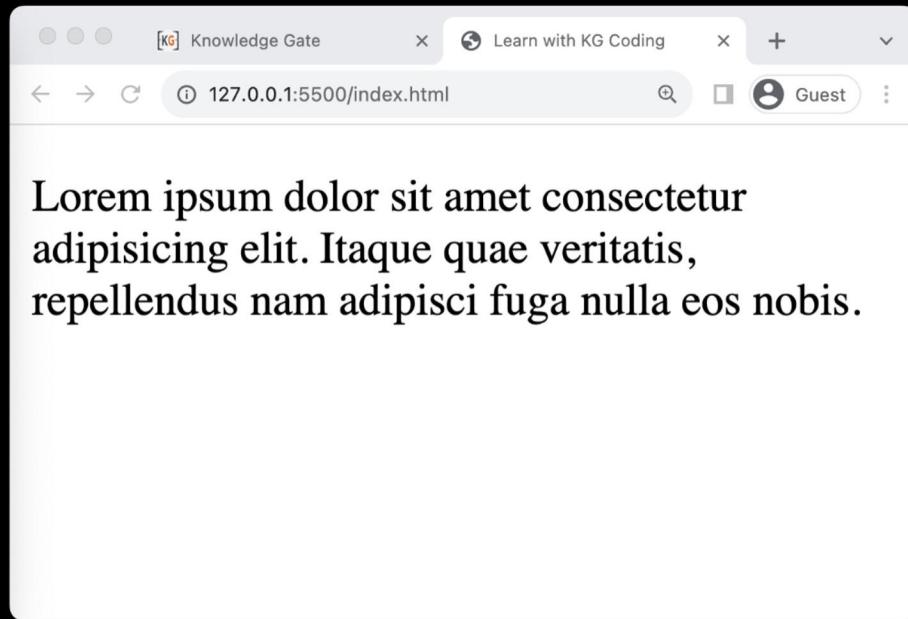
1. Used for creating **hyperlinks**
2. Requires **href** attribute for URL
3. Can link to external sites or internal pages
4. Supports **target** attribute to control link behavior

# 14 Div Tags

The screenshot shows a code editor interface with a sidebar containing icons for file, search, and other functions. The main area displays the following code:

```
3 <head>
4   <title>Learn with KG Coding</title>
5 </head>
6 <body>
7   <div>
8     <p> Lorem ipsum dolor sit amet
       consectetur adipisicing elit. Itaque quae
       veritatis, repellendus nam adipisci fuga
       nulla eos nobis.</p>
9   </div>
10 </body>
11 </html>
```

A DOM tree is visible above the code, showing the structure: index.html > html > body > div > p.

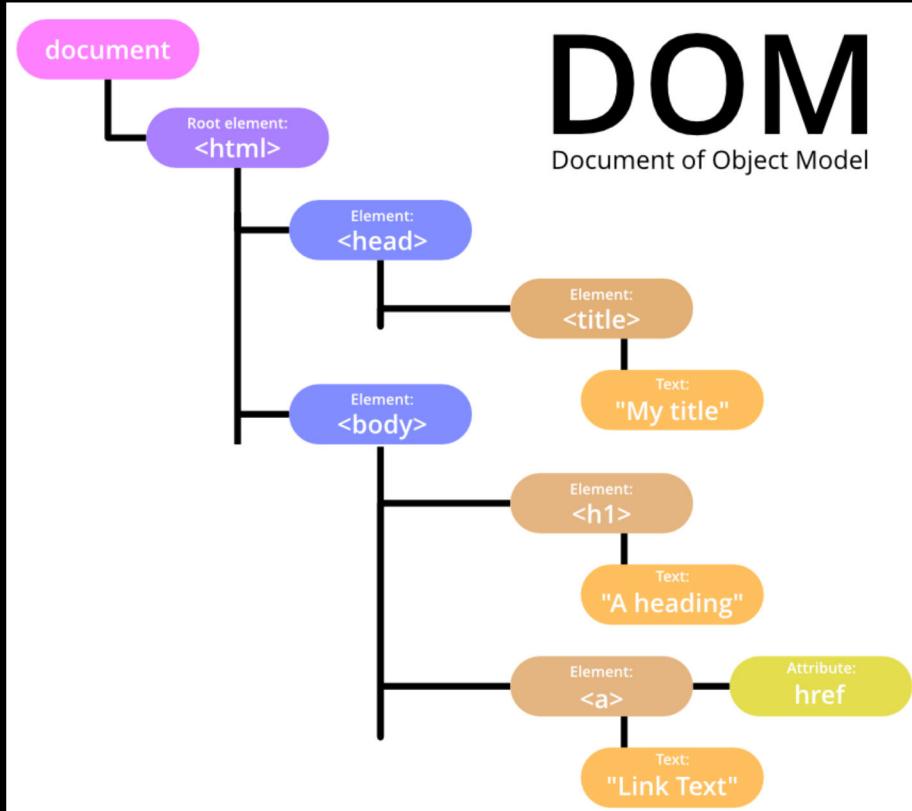


1. **Purpose:** Acts as a container for other **HTML** elements.
  2. **Non-Semantic:** Doesn't provide inherent meaning to enclosed content.
  3. **Styling:** Commonly used for layout and styling via **CSS**.
  4. **Flexibility:** Highly versatile and can be customized using classes or IDs.

# 14 Basic HTML Page

```
<!DOCTYPE html>           Defines the HTML Version  
  
<html lang="en">          Parent of all HTML tags / Root element  
  
    <head>                  Parent of meta data tags  
        <title>My First Webpage</title>  Title of the web page  
  
    </head>  
  
    <body>                  Parent of content tags  
        <h1>Hello World!</h1>  Heading tag  
  
    </body>  
  
</html>
```

# 14 HTML DOM



- 1. Structure Understanding:** Helps in understanding the **hierarchical structure** of a webpage, crucial for applying targeted CSS styles.
- 2. Dynamic Styling:** Enables learning about dynamic styling, allowing for **real-time changes** and interactivity through CSS.

# 14 HTML (Attributes)

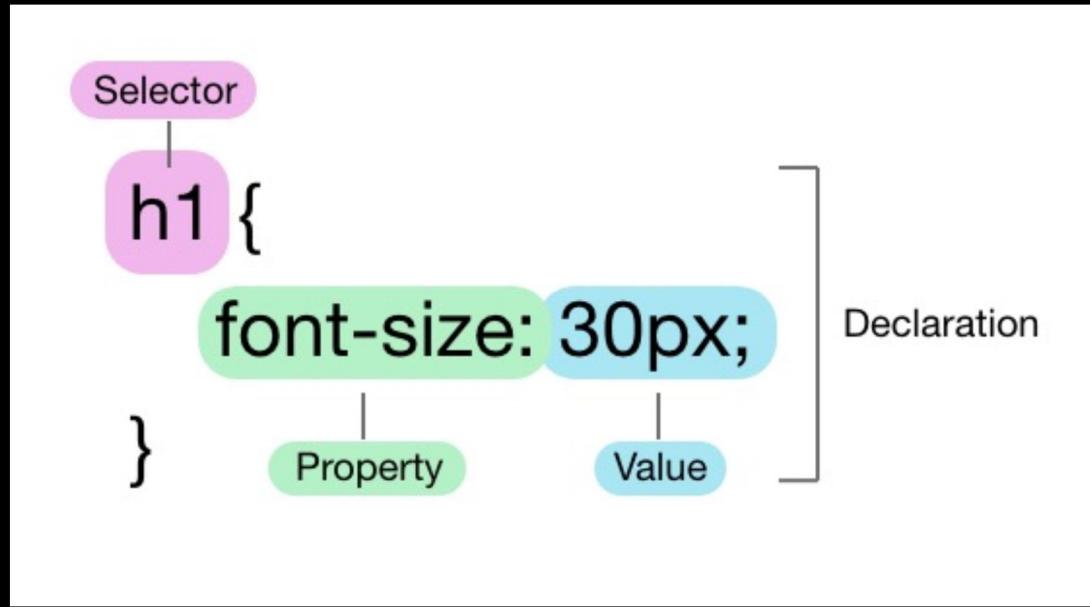
## Html Attributes

Attribute

```
<tag attribute="value">Text Content </tag>
```

1. Provides **additional information** about elements
2. Placed **within opening tags**
3. Common examples: **href, src, alt**
4. Use **name=value** format
5. Can be **single or multiple** per element

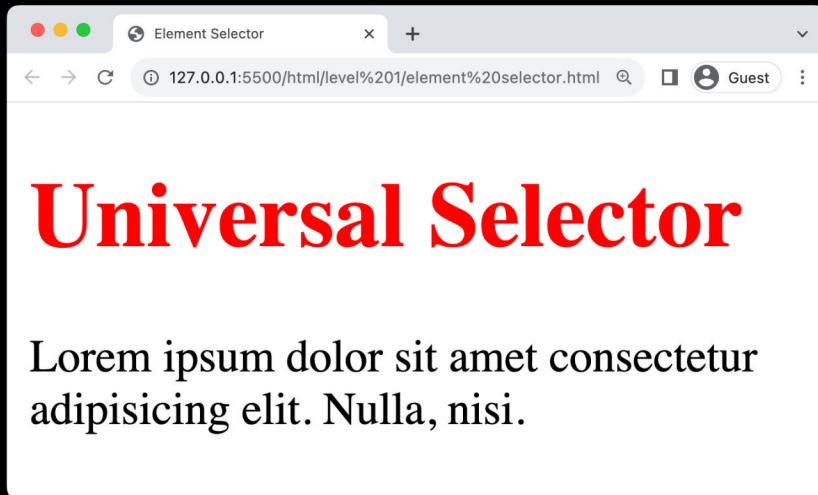
# 15 CSS (Basic Syntax)



- **Selector**: The HTML element that you want to style.
- **Property**: The attribute you want to change (like font, color, etc.).
- **Value**: The specific style you want to apply to the property (like red, bold, etc.).

# 15 CSS (Element selector)

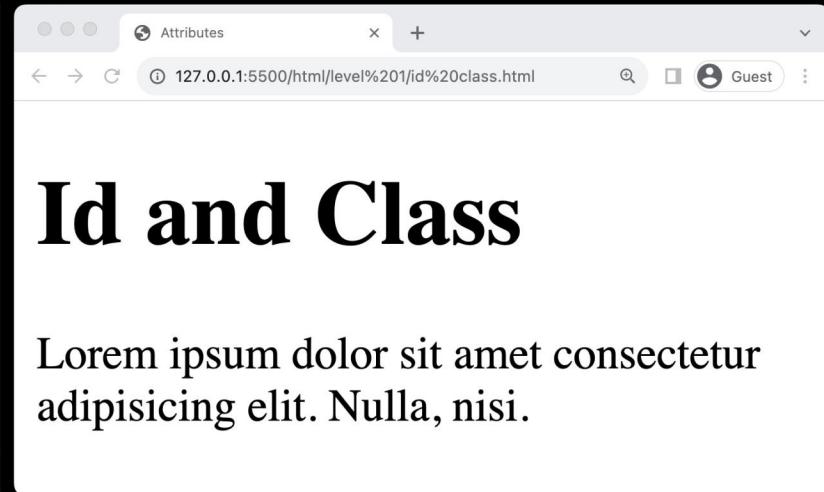
```
3 <head>
4     <title>Element Selector</title>
5     <style>
6         h1 {
7             color: red
8         }
9     </style>
10    </head>
11    <body>
12        <h1>Universal Selector</h1>
13        <p>Lorem ipsum dolor sit amet consectetur
14            adipisicing elit. Nulla, nisi.</p>
```



- **Targets Elements:** Selects HTML elements based on their **tag name**.
- **Syntax:** Simply use the **element's name**
- **Uniform Styling:** Helps in applying **consistent styles** to all instances.
- **Ease of Use:** Straightforward and **easy** to implement for basic styling.

# 15 CSS (id & class attribute)

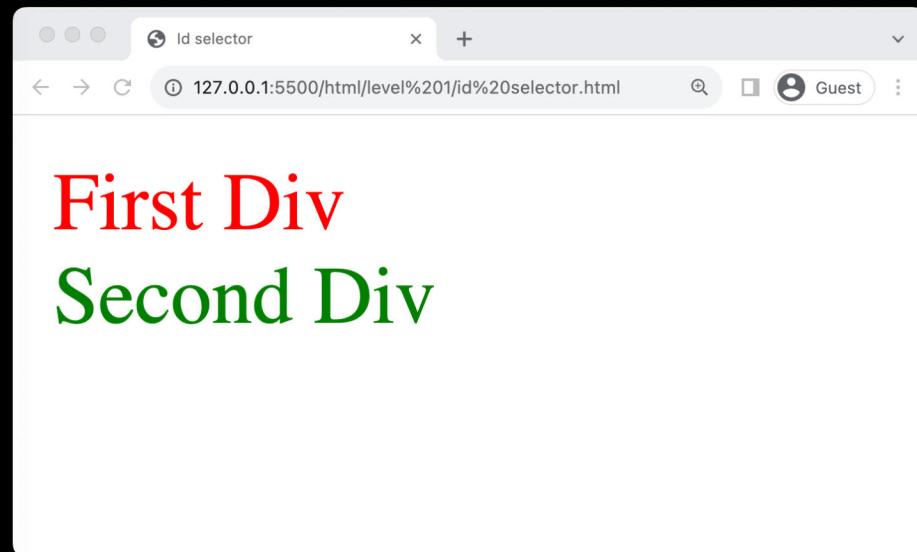
```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4  |   <title>Attributes</title>
5  </head>
6  <body>
7  |   <h1 id="top_heading">Id and Class</h1>
8  |   <p class="article">Lorem ipsum dolor sit amet
9   |       consectetur adipisicing elit. Nulla, nisi.</p>
9  </body>
10 </html>
```



- **ID Property:** Assigns a unique identifier to a single HTML element.
- **Class Property:** Allows grouping of multiple HTML elements to style them collectively.
- **Reusable Classes:** Class properties can be reused across different elements for consistent styling.
- **Specificity and Targeting:** Both properties assist in targeting specific elements or groups of elements for precise styling.

# 15 CSS (Id selector)

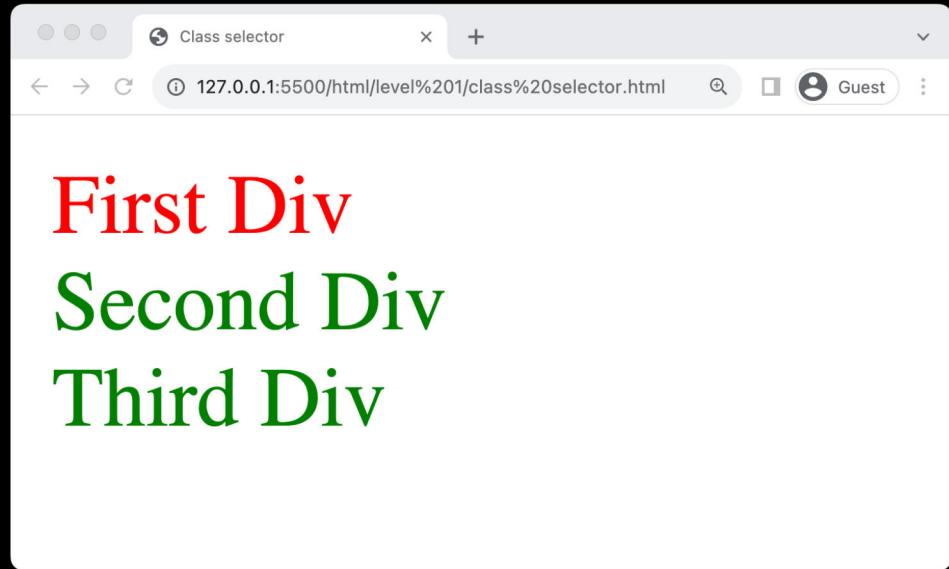
```
3 <head>
4     <title>Id selector</title>
5     <style>
6         #first { color: red; }
7         #second { color: green; }
8     </style>
9 </head>
10 <body>
11     <div id="first">First Div</div>
12     <div id="second">Second Div</div>
13 </body>
```



- **Unique Identifier:** Targets a specific element with a **unique ID** attribute.
- **Syntax:** Uses the **hash (#)** symbol
- **Single Use:** Each ID should be used **once per page** for uniqueness.
- **Specific Targeting:** Ideal for styling **individual, distinct** elements.

# 15 CSS (Class selector)

```
<head>
    <title>Class selector</title>
    <style>
        #first { color: red; }
        .second { color: green; }
    </style>
</head>
<body>
    <div id="first">First Div</div>
    <div class="second">Second Div</div>
    <div class="second">Third Div</div>
</body>
```

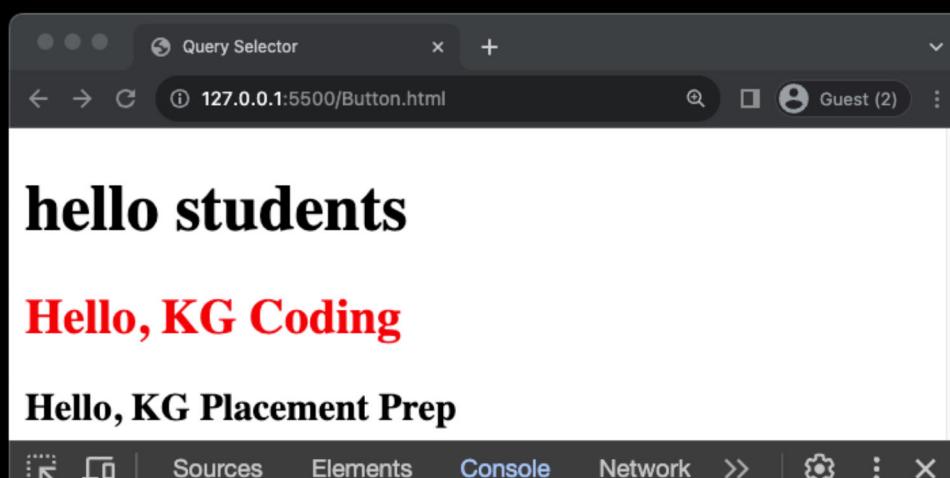
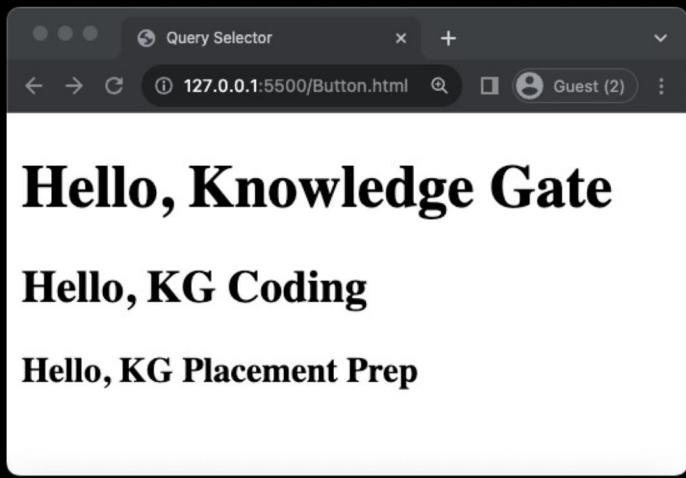


- **Group Styling:** Allows styling of **multiple elements** grouped under a class.
- **Syntax:** Utilizes the **dot (.)** symbol.
- **Reusable:** Can be used on multiple elements for **consistent styling**.
- **Versatility:** Ideal for applying styles to a category of elements.

# 16 Query Selector

1. **getElementById**: Finds one element by its ID.
2. **getElementsByClassName**: Finds elements by their class, returns a list.
3. **querySelector**: Finds the first element that matches a CSS selector.
4. **Purpose**: To interact with or modify webpage elements.

# 16 Query Selector



```
Button.html — testing
Button.html ×

<!DOCTYPE html>
<html lang="en">
  <head>
    <title>Query Selector</title>
  </head>
  <body>
    <h1>Hello, Knowledge Gate</h1>
    <h2 class="coding">Hello, KG Coding</h2>
    <h3 id="placement">Hello, KG Placement Prep</h3>
  </body>
</html>
```

```
const hElement = document.querySelector('h1');
const hElementCode = document.querySelector('.coding');
const hElementPlacement =
document.querySelector('#placement');

hElement.textContent = 'hello students';
hElementCode.style.color = 'red';

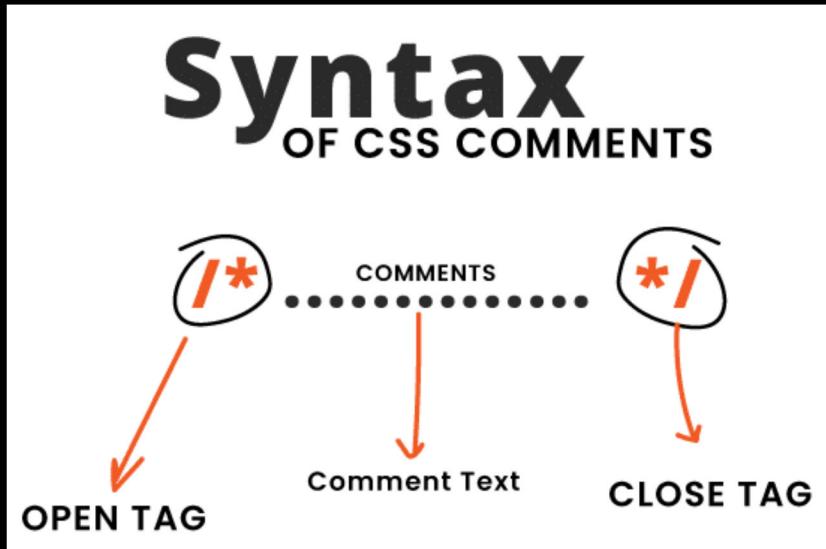
-> 'red'
```

# 17 Script Tag

1. **Embed Code:** Incorporates JavaScript into an HTML file, either **directly or via external files.**
2. **Placement:** Commonly placed in the `<head>` or just before the closing `</body>` tag to control when the script runs.
3. **External Files:** Use `src` attribute to link external JavaScript files, like `<script src="script.js"></script>`.
4. **Console Methods:** `log`, `warn`, `error`, `clear`

# 18 JavaScript & CSS Comments

1. Used to add notes in source code in JavaScript or CSS.
2. Not displayed on the web page
3. Syntax: `/* comment here */`
4. Helpful for code organization
5. Can be multi-line or single-line



# 18 HTML Comments

1. Used to add **notes** in HTML code
2. **Not displayed** on the web page
3. Syntax: `<!-- Comment here -->`
4. Helpful for **code organization**
5. Can be **multi-line** or **single-line**

Writing comments in HTML

Single-line  
Comment

```
1 <!--This is a single line  
comment in HTML. You cannot  
see it on a webpage. Click  
on view-source to see a  
message I left just for you.  
-->
```

```
1 <!-- This is a multi-line  
comment in HTML.  
2 You cannot see it on a  
webpage.  
3 If you view-source on the  
browser you can see the  
comment there.-->
```

Multi-line  
Comment

# JavaScript with HTML & CSS Revision

13. VS Code IDE

14. HTML Tags Introduction

15. CSS Introduction

16. Query Selector

17. Script Tag

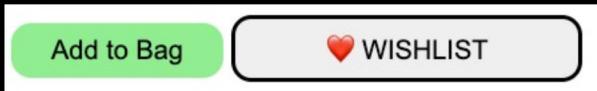
18. Comments



# Practice Exercise

## JavaScript With HTML & CSS

1. Create a **button** with text click
2. Create 2 buttons with **class** and **id**
3. Create a **paragraph**
4. Add **colors** to two buttons
5. Add proper html **structure**
6. Change page **title**
7. Try to copy the given design on the bottom



8. Add **script element** to page to **show welcome alert**
9. Add **onclick** alert on **Add to Bag** and **Heart Wishlist** buttons

# KG Coding

Some Other One shot Video Links:

- [Complete HTML](#)
- [Complete CSS](#)
- [Complete JavaScript](#)
- [Complete React and Redux](#)
- [One shot University Exam Series](#)



<http://www.kgcoding.in/>

Our YouTube Channels

[KG Coding Android App](#)



[KG Coding](#)



[Knowledge GATE](#)



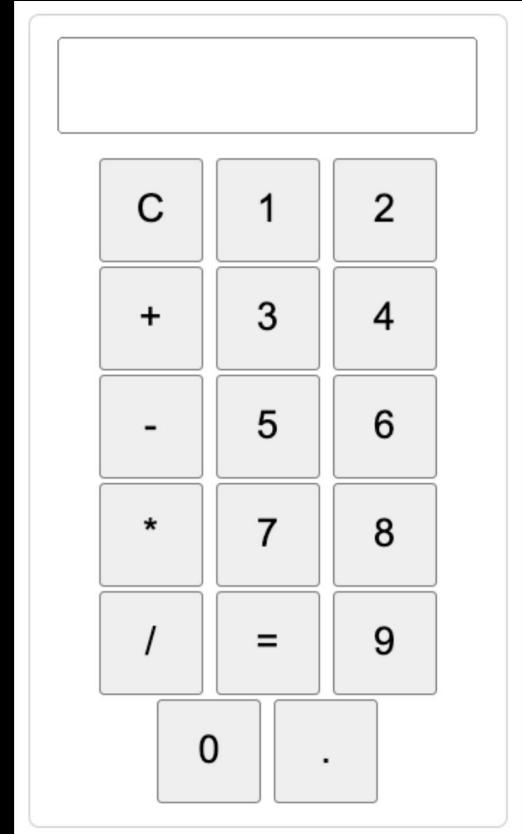
[KG Placement Prep](#)



[Sanchit Socket](#)

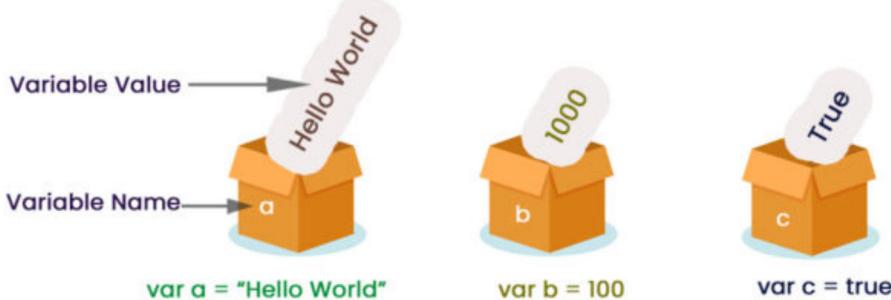
# Variables

19. What are Variables
20. Syntax Rules
21. Updating Values
22. Myntra Bag Exercise
23. Naming Conventions
24. Ways to create Variables



# 19. What are Variables?

Variable is used to Store Data



Variables are like containers  
used for storing data values.

# 20. Syntax Rules

```
1 // Defining a number variable  
2 let noOfStudents = 5;  
3 // Defining a String variable  
4 let welcomeMessage = "Hello Beta"
```

1. Can't use **keywords** or reserved words
2. Can't start with a **number**
3. No special characters other than **\$** and **\_**
4. **=** is for **assignment**
5. **:** means end of **instruction**

# 21. Updating Values

```
let noOfStudents = 5;  
noOfStudents = noOfStudents + 1;  
  
let money = 1;  
money += 5; // money = 6  
money -= 2; // money = 4  
money *= 3; // money = 12  
money /= 4; // money = 3  
money++; // money = 4
```

1. Do not need to use **let** again.
2. Syntax: **variable = variable + 1**
3. Assignment Operator is used **=**
4. Short Hand Assignment Operators:  
**+ =, - =, \* =, / =, ++**

# 22. Myntra Bag Exercise

Add to Bag

MOVE TO WISHLIST

Add 1+1 sale item

Your Bag has 2 items

# 23. Naming Conventions

## camelCase

- Start with a lowercase letter. Capitalize the first letter of each subsequent word.
- Example: `myVariableName`



camelCase

## snake\_case

- Start with an lowercase letter. Separate words with **underscore**.
- Example: `my_variable_name`



snake\_case

## Kebab-case

- All lowercase letters. Separate words with **hyphens**. Used for HTML and CSS.
- Example: `my-variable-name`



kebab-case

## Keep a Good and Short Name

- Choose names that are descriptive but not too long. It should make it easy to understand the variable's purpose.
- Example: `age`, `firstName`, `isMarried`

# 24. Ways to Create Variables

var

var apple = 



a thing in a box  
named "apple"

apple = 



you can swap  
item later

const

let

var

let

let apple = 



a thing in a box  
named "apple" w/  
protection shield

apple =  NG

ok!  
you can swap item  
only if you ask  
inside of the shield

const

const apple = 



a thing in  
LOCKED cage  
named "apple"



apple =  NG

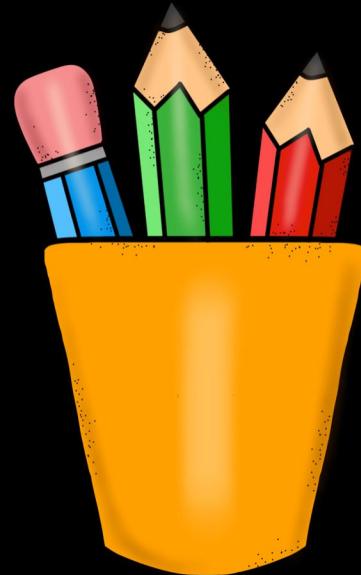
you can't  
swap item  
later.



apple.multiply(3)  
... but you can ask  
the item to change itself  
(if the item has method  
to do that)

# Variables Revision

- 19. What are Variables
- 20. Syntax Rules
- 21. Updating Values
- 22. Myntra Bag Exercise
- 23. Naming Conventions
- 24. Ways to create Variables



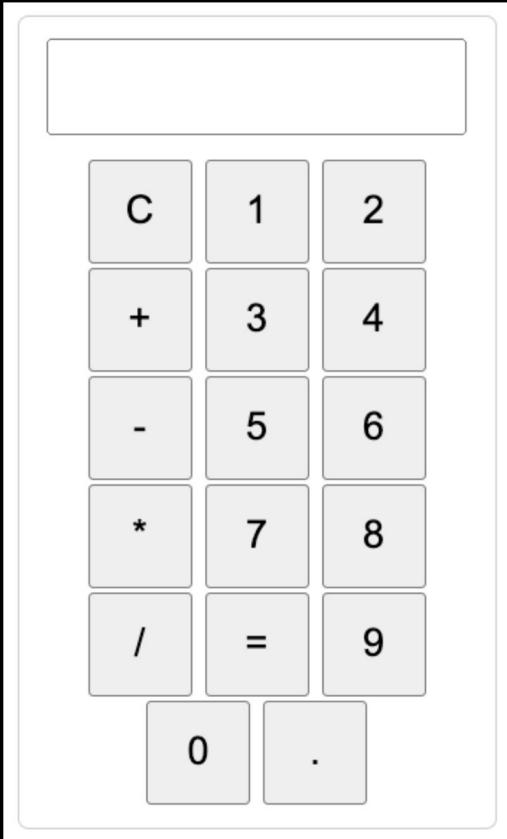
# Practice Exercise

## Variables

1. Save your name in a variable inside script tag
2. Display name from the variable on the page
3. Calculate the cost of Myntra Bag and keep it in a variable
4. Show it to console
5. Keep GST percentage as constant
6. Use eval method from math to convert string calculation into result



# Project One: Calculator



# KG Coding

Some Other One shot Video Links:

- [Complete HTML](#)
- [Complete CSS](#)
- [Complete JavaScript](#)
- [Complete React and Redux](#)
- [One shot University Exam Series](#)



<http://www.kgcoding.in/>

Our YouTube Channels

[KG Coding Android App](#)



[KG Coding](#)



[Knowledge GATE](#)



[KG Placement Prep](#)



[Sanchit Socket](#)

# if-else & Boolean

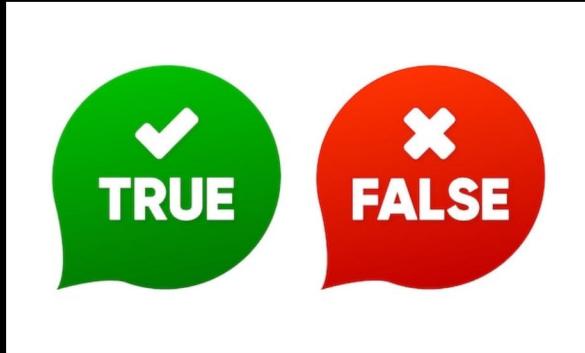
25. What are Booleans
26. Comparison Operators
27. if-else
28. Logical Operator
29. Scope
30. Truthy and Falsy Values
31. If alternates

Create



Project

# 25. What are Booleans?



1. Data Type: Booleans are a basic data type in JavaScript.
2. Two Values: Can only be true or false.
3. 'true' is a String not a Boolean

# 26. Comparison Operators

< , > , <= , >= , == , !=

## Equality

== Checks value equality.

=== Checks value and type equality.

## Inequality

!= Checks value inequality.

!== Checks value and type inequality.

## Relational

> Greater than.

< Less than.

>= Greater than or equal to.

<= Less than or equal to.

Order of comparison operators is less than arithmetic operators

# 27. if-else

1. **Syntax:** Uses `if () {}` to check a condition.
2. **What is if:** Executes block if condition is `true`, skips if `false`.
3. **What is else:** Executes a block when the if condition is `false`.
4. **Curly Braces** can be omitted for single statements, but not recommended.
5. **If-else Ladder:** Multiple if and else if blocks; **only one** executes.
6. **Use Variables:** Can store **conditions** in variables for use in if statements.

```
if thirsty {
```



```
} else {
```



```
}
```

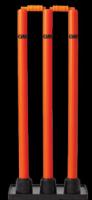
# Project Cricket Game



*Defeats*



*Defeats*



*Defeats*



## Bat Ball Stump Game

Bat

Ball

Stump

# 28. Logical Operators



AND

Or

not

1. Types: **&&** (AND), **||** (OR), **!** (NOT)
2. AND (**&&**): All conditions **must be true** for the result to be true.
3. OR (**||**): Only **one condition** must be true for the result to be true.
4. NOT (**!**): **Inverts** the Boolean value of a condition.
5. Lower Priority than **Math** and **Comparison** operators

# Project Cricket Game



*Defeats*



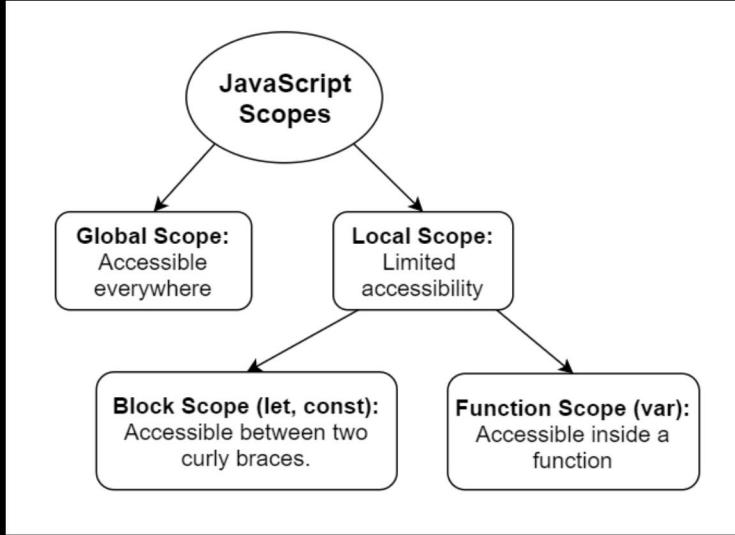
*Defeats*



*Defeats*



# 29. Scope



1. Any **variable** created inside `{}` will remain inside `{}`
2. **Variable** can be **redefined** inside `{}`
3. **Var** does **not** follow **scope**
4. Global Scope: Accessible **everywhere** in the code.
5. Block Scope: **Limited** to a block, mainly with **let** and **const**.
6. Declare variables in the **narrowest** scope possible.

# Project Cricket Game



*Defeats*



*Defeats*



*Defeats*



# 30. Truthy and Falsy Values

1. Falsy Values: 0, null, undefined, false, NaN, "" (empty string)
2. Truthy Values: All values **not** listed as falsy.
3. Used in **conditional** statements like **if**.
4. **Non-boolean** values are **auto-converted** in logical operations.
5. Be **explicit** in **comparisons** to avoid unexpected behaviour.



# 31. If alternates

1. Ternary Operator: `condition ? trueValue : falseValue`

Quick one-line if-else.

2. Guard Operator: `value || defaultValue`

Use when a `fallback` value is needed.

3. Default Operator: `value ?? fallbackValue`

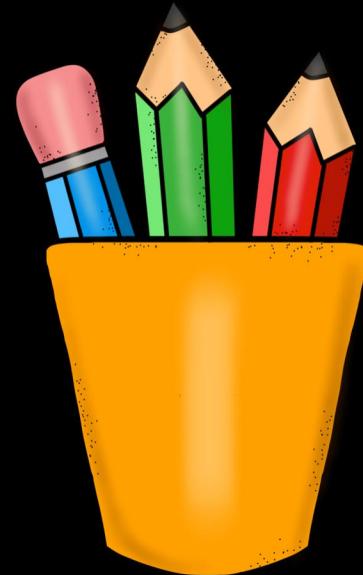
Use when you want to consider only null and undefined as falsy.

4. Simplifies conditional logic.

5. Use wisely to maintain readability.

# if-else & Boolean Revision

- 25. What are Booleans
- 26. Comparison Operators
- 27. if-else
- 28. Logical Operator
- 29. Scope
- 30. Truthy and Falsy Values
- 31. If alternates



# Practice Exercise

if-else & Boolean

1. Give discount based on age, gender for metro ticket

- Females get 50% off
- Kids under 5 years of age are free
- Kids up to 8 years have half ticket
- People Over 65 years of age only pay 30% of the ticket
- In case of multiple discounts max discount will apply.



# KG Coding

Some Other One shot Video Links:

- [Complete HTML](#)
- [Complete CSS](#)
- [Complete JavaScript](#)
- [Complete React and Redux](#)
- [One shot University Exam Series](#)



<http://www.kgcoding.in/>

Our YouTube Channels

[KG Coding Android App](#)



[KG Coding](#)



[Knowledge GATE](#)



[KG Placement Prep](#)



[Sanchit Socket](#)

# Functions

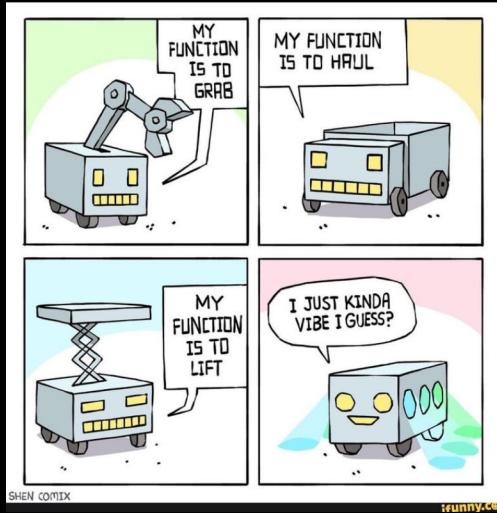
- 32. What are Functions
- 33. Function Syntax
- 34. Return statement
- 35. Function Parameters

Improve



Project

# 32. What are Functions?

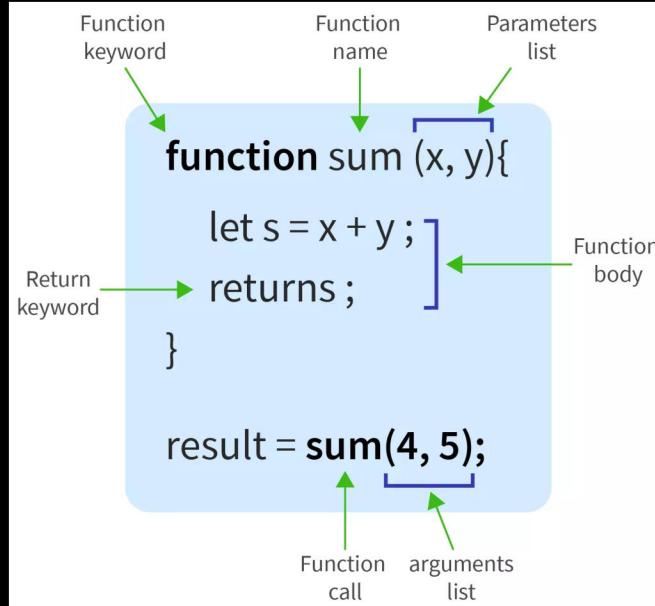


```
function greet(name) {  
    // code  
}  
  
greet(name);  
// code
```

A diagram showing a function definition and a call to it. The function is defined as `function greet(name) { }` , with the body containing `// code`. An arrow points from the word `greet` in the definition to the same word in the call `greet(name);`, which is preceded by another `// code` comment. This illustrates how a single function can be reused in different parts of a program.

1. Definition: Blocks of **reusable** code.
2. DRY Principle: "**Don't Repeat Yourself**" it Encourages code reusability.
3. Usage: Organizes **code** and performs specific tasks.
4. Naming Rules: Same as **variable** names: **camelCase**
5. Example: "Beta Gas band kar de"

# 33. Functions Syntax



1. Use **function keyword** to declare.
2. Follows same rules as **variable names**.
3. Use **()** to contain parameters.
4. Invoke by using the **function name** followed by **()**.
5. Fundamental for **code organization** and **reusability**.

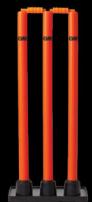
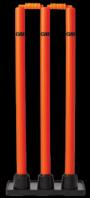
# Project Cricket Game



*Defeats*



*Defeats*



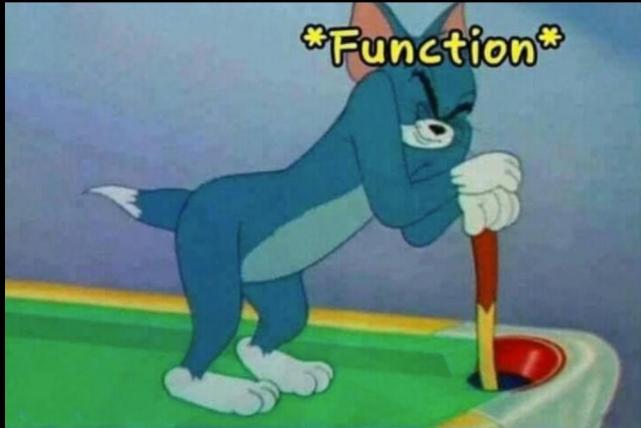
*Defeats*



**Create Method for Random Number & Computer Choice**



# 34. Return statement



1. Sends a value back from a function.
2. Example: "Ek glass paani laao"
3. What Can Be Returned: Value, variable, calculation, etc.
4. Return ends the function immediately.
5. Function calls make code jump around.
6. Prefer returning values over using global variables.

# Project Cricket Game



*Defeats*



*Defeats*



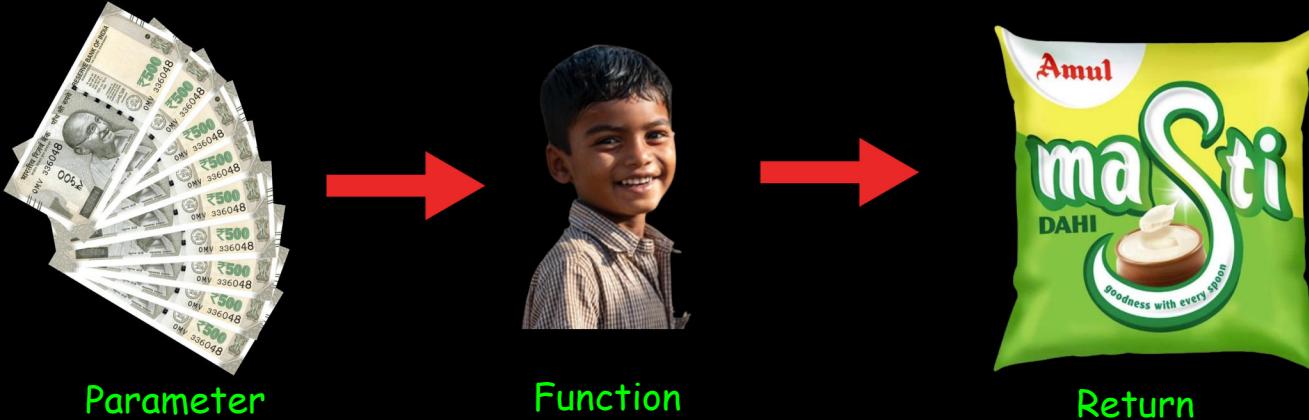
*Defeats*



User return instead of Global Variable



# 35. Parameters



1. Input values that a **function** takes.
2. Parameters put value into function, while return gets value out.
3. Example: "Ek packet dahi laao"
4. Naming Convention: Same as **variable** names.
5. **Parameter vs Argument**
6. Examples: **alert**, **Math.round**, **console.log** are functions we have already used
7. **Multiple Parameters:** Functions can take **more than one**.
8. **Default Value:** Can set a **default** value for a parameter.

# Project Cricket Game



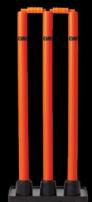
*Defeats*



Create method for comparing user choice & Showing Result alert



*Defeats*

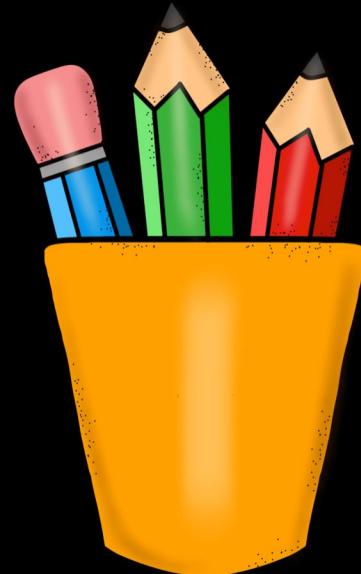


*Defeats*



# Functions Revision

- 32. What are Functions
- 33. Function Syntax
- 34. Return statement
- 35. Function Parameters



# Practice Exercise

## Functions

1. Create a method to check if a number is odd or even.
2. Create a method to return larger of the two numbers.
3. Create Method to convert Celsius to Fahrenheit  
 $F = (9/5) * C + 32$



# KG Coding

Some Other One shot Video Links:

- [Complete HTML](#)
- [Complete CSS](#)
- [Complete JavaScript](#)
- [Complete React and Redux](#)
- [One shot University Exam Series](#)



<http://www.kgcoding.in/>

Our Channels

KG Coding Android App



[KG Coding](#)



[Knowledge GATE](#)



[KG Placement Prep](#)



[Sanchit Socket](#)

# Objects

- 36. What is an Object
- 37. Objects Syntax
- 38. Accessing Objects
- 39. Inside Objects
- 40. Autoboxing
- 41. Object References
- 42. Object Shortcuts

Maintain score of



Project

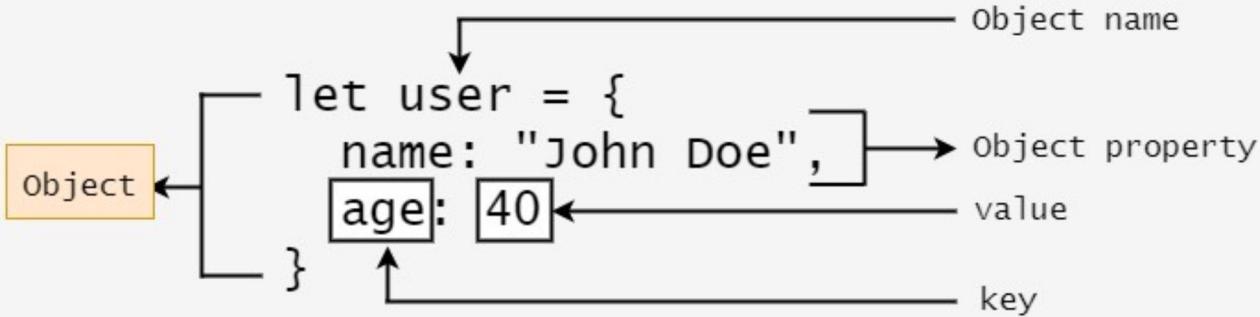
# 36. What is an Object?



```
let product = {  
    company: 'Mango',  
    item_name: 'Cotton striped t-shirt',  
    price: 861  
};
```

1. Groups multiple **values** together in **key-value** pairs.
2. How to Define: Use **{}** to enclose **properties**.
3. Example: **product {name, price}**
4. Dot Notation: Use **. operator** to access values.
5. Key Benefit: Organizes **related data** under a **single name**.

# 37. Object Syntax?



1. Basic Structure: Uses `{}` to enclose data.
2. Rules: **Property** and **value** separated by a **colon(:)**
3. Comma: Separates different **property-value** pairs.
4. Example: `{ name: "Laptop", price: 1000 }`

# 38. Accessing Objects



1. Dot Notation: Access **properties** using **.** Operator like `product.price`
2. Bracket Notation: Useful for **properties** with **special characters** `product["nick-name"]`. Variables can be used to access properties
3. **typeof** returns **object**.
4. Values can be **added** or **removed** to an object
5. Delete Values using **delete**

# Project Cricket Game



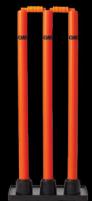
*Defeats*



Create object for maintaining Score



*Defeats*



*Defeats*



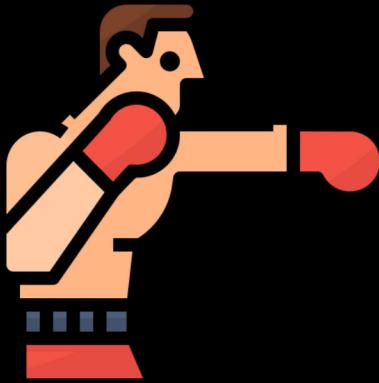
# 39. Inside Object

```
let product = {  
    company: 'Mango',  
    itemName: 'Cotton striped t-shirt',  
    price: 861,  
    rating: {  
        stars: 4.5,  
        noOfReviews: 87  
    },  
    displayPrice: function() {  
        return `$$ {this.price.toFixed(2)} `;  
    }  
};
```



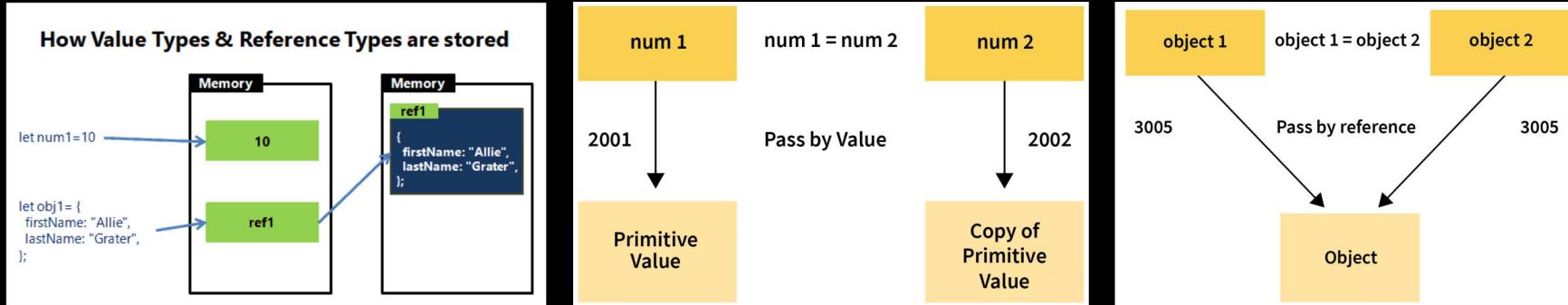
1. Objects can contain **Primitives** like **numbers** and **strings**.
2. Objects can contain **other objects** and are called **Nested Objects**.
3. **Functions** can be object properties.
4. Functions inside an object are called **methods**.
5. Null Value: **Intentionally** leaving a property **empty**.

# 40. Autoboxing



1. Automatic conversion of primitives to objects.
2. Allows properties and methods to be used on primitives.
3. Example: Strings have properties and methods like length, toUpperCase, etc.

# 41. Object References



1. Objects work based on **references**, not actual data.
2. Copying an object copies the reference, not the actual object.
3. When comparing with `==`, you're comparing references, not content.
4. Changes to one reference affects all copies.

# 42. Object Shortcuts

```
let product = {  
    company: 'Mango',  
    itemName: 'Cotton striped t-shirt',  
    price: 861  
};  
  
// Destructuring  
let company = product.company  
// is same as  
let { company } = product;
```

```
// Property shorthand  
let price = 861;  
let product = {  
    company: 'Mango',  
    itemName: 'Cotton striped t-shirt',  
    price: price  
};  
  
// is same as  
let product1 = {  
    company: 'Mango',  
    itemName: 'Cotton striped t-shirt',  
    price  
};
```

```
// Method shorthand  
let product = {  
    company: 'Mango',  
    itemName: 'Cotton striped t-shirt',  
    displayPrice: function() {  
        return `$$this.price.toFixed(2)`;  
    }  
};  
  
// is same as  
let product1 = {  
    company: 'Mango',  
    itemName: 'Cotton striped t-shirt',  
    displayPrice() {  
        return `$$this.price.toFixed(2)`;  
    }  
};
```

1. De-structuring: Extract properties from objects easily.
2. We can extract more than one property at once.
3. Shorthand Property: {message: message} simplifies to just message.
4. Shorthand Method: Define methods directly inside the object without the function keyword.

# Project Cricket Game



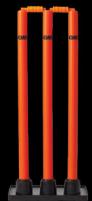
*Defeats*



Move method for showing result into score  
Object.



*Defeats*

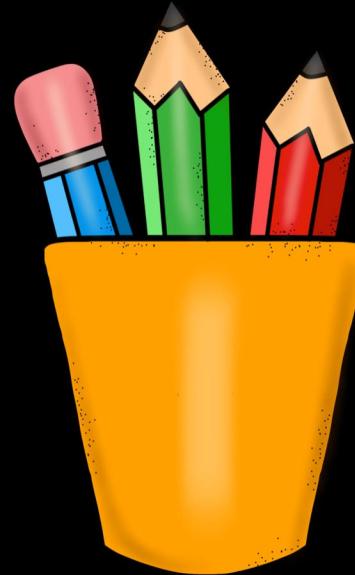


*Defeats*



# Objects Revision

- 36. What is an Object
- 37. Objects Syntax
- 38. Accessing Objects
- 39. Inside Objects
- 40. Autoboxing
- 41. Object References
- 42. Object Shortcuts



# Practice Exercise

## Objects

1. Create **object** to represent a **product** from **Myntra**
2. **Create an Object** with **two references** and log changes to one object by **changing** the other one.
3. Use bracket notation to display **delivery-time**.
4. **Given** an object `{message: 'good job', status: 'complete'}`, use de-structuring to create two variables **message** and **status**.
5. Add **function** `isIdenticalProduct` to compare two product objects.



# KG Coding

Some Other One shot Video Links:

- [Complete HTML](#)
- [Complete CSS](#)
- [Complete JavaScript](#)
- [Complete React and Redux](#)
- [One shot University Exam Series](#)



<http://www.kgcoding.in/>

Our YouTube Channels

[KG Coding Android App](#)



[KG Coding](#)



[Knowledge GATE](#)



[KG Placement Prep](#)



[Sanchit Socket](#)

# JSON, Local Storage, Date & DOM

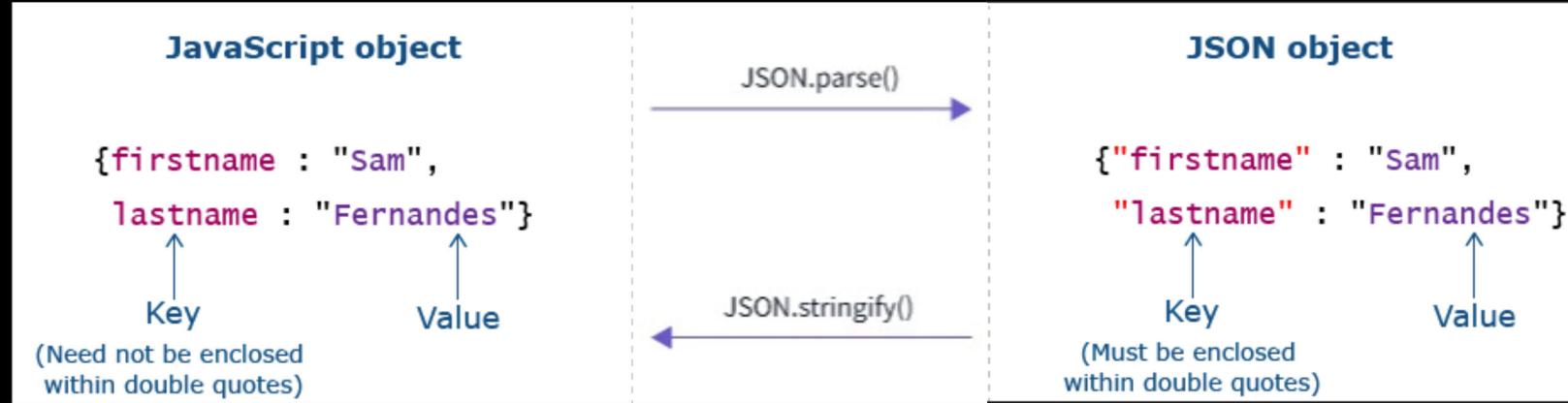
- 43. What is JSON?
- 44. Local Storage
- 45. Date
- 46. DOM Properties & Methods

Finish



Project

# 43. What is JSON?

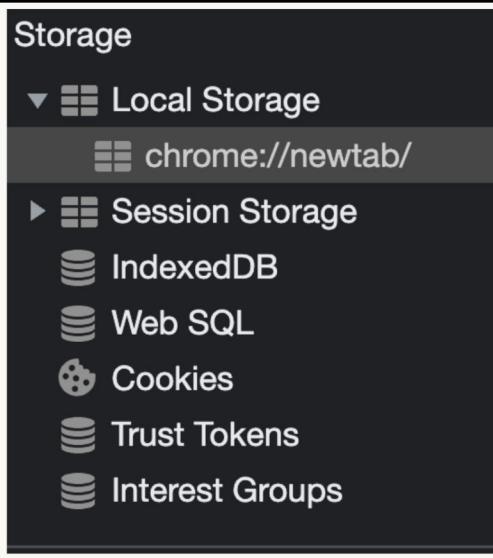


1. **JavaScript Object Notation:** Not the same as JS object, but similar.
2. Common in network calls and data storage.
3. **JSON.stringify()** and **JSON.parse()**
4. Strings are easy to transport over network.
5. JSON requires double quotes. Escaped as \".
6. JSON is data format, JS object is a data structure.

# 44. Local Storage

```
// store an object in Local Storage
localStorage.setItem(
  "user",
  JSON.stringify({
    name: "Gopi Gorantala"
    age: 32
  });
);

// retrieve an object in Local Storage
const user = JSON.parse(
  localStorage.getItem("user")
);
```



1. Persistent data storage in the browser.
2. `setItem`: Stores data as key-value pairs.
3. Only strings can be stored.
4. `getItem`: Retrieves data based on key.
5. Other Methods: `localStorage.clear()`, `removeItem()`.
6. Do not store sensitive information. Viewable in storage console.

# Project Cricket Game



*Defeats*



*Defeats*



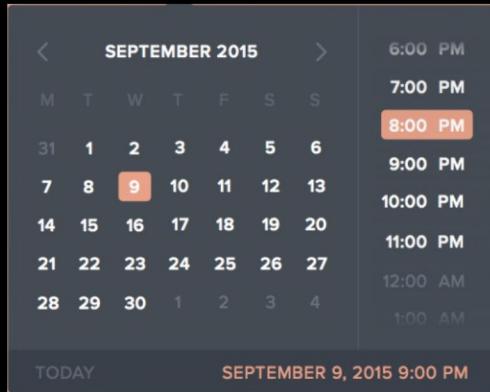
*Defeats*



- 1. Score will survive browser refresh.
- 2. Add Reset Button To clear or reset stored data.



# 45. Date



1. new Date() Creates a new Date object with the current date and time.
2. Key Methods:
  - `getTime()`: Milliseconds since Epoch.
  - `getFullYear()`: 4-digit year
  - `getDay()`: Day of the week
  - `getMinutes()`: Current minute
  - `getHours()`: Current hour.
3. Crucial for timestamps, scheduling, etc.

# 46. DOM Properties & Methods

## DOM and Element Properties

- 1. location
- 2. title
- 3. href
- 4. domain
- 5. innerHTML
- 6. innerText
- 7. classList

## DOM and Element Methods

- 1. getElementById()
- 2. querySelector()
- 3. classList: add(), remove()
- 4. createElement()
- 5. appendChild()
- 6. removeChild()
- 7. replaceChild()

# Project Cricket Game



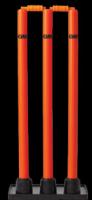
*Defeats*



1. Show moves in the page instead of the alert
2. Show result in the page instead of the alert



*Defeats*



*Defeats*



# Project Cricket Game



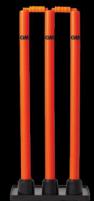
*Defeats*



1. Replace the Bat-Ball-Stump Buttons with Images



*Defeats*



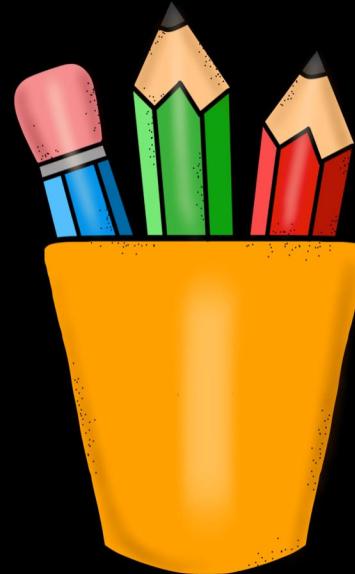
*Defeats*



# JSON, Local Storage, Date & DOM

## Revision

- 43. What is JSON?
- 44. Local Storage
- 45. Date
- 46. DOM Properties & Methods



# Practice Exercise

## JSON ,Local Storage, Date & DOM

1. Display good morning, afternoon and night based on current hour.
2. Add the name to the output too.
3. Create a Button which shows the number how many times it has been pressed.
  - Also, it has different colors for when it has been pressed odd or even times.
  - The click count should also survive browser refresh.



# KG Coding

Some Other One shot Video Links:

- [Complete HTML](#)
- [Complete CSS](#)
- [Complete JavaScript](#)
- [Complete React and Redux](#)
- [One shot University Exam Series](#)



<http://www.kgcoding.in/>

Our Channels

KG Coding Android App



[KG Coding](#)



[Knowledge GATE](#)



[KG Placement Prep](#)

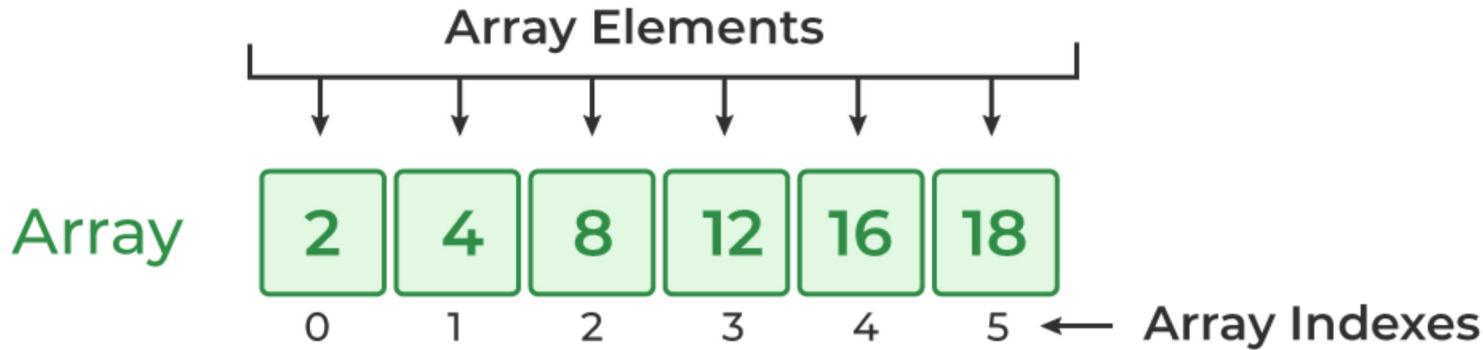


[Sanchit Socket](#)

# Arrays & Loops

- 47. What is an Array?
- 48. Array Syntax & Values
- 49. Array Properties & Methods
- 50. What is a Loop?
- 51. While Loop
- 52. Do While Loop
- 53. For Loop
- 54. Accumulator Pattern
- 55. Break & Continue

# 47. What is an Array?



1. An Array is just a **list** of values.
2. Index: Starts with **0**.
3. Arrays are used for **storing multiple values** in a single variable.

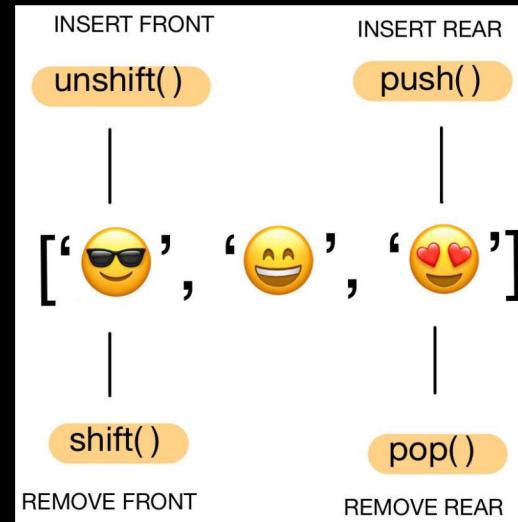
# 48. Array Syntax & Values

```
let myArray = [1, 'KG Coding', null, true,  
  {likes: '1 Million'}];
```

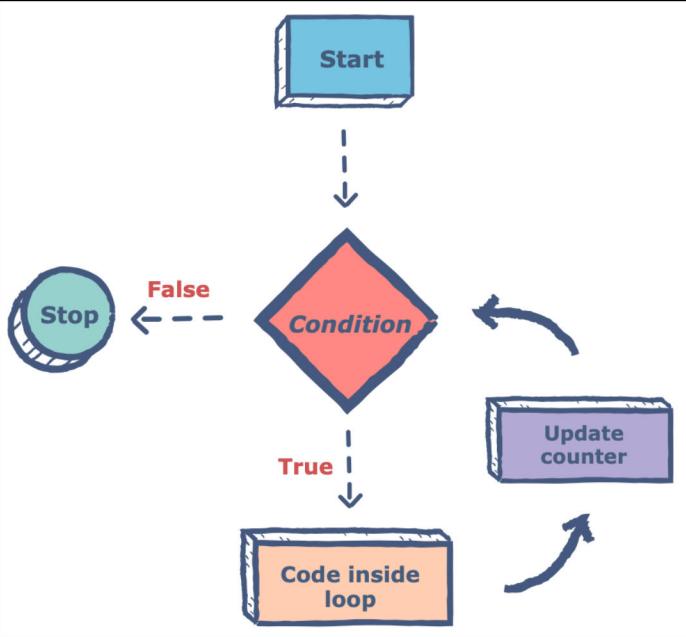
1. Use [] to create a new array, [] brackets enclose list of values
2. Arrays can be saved to a variable.
3. Accessing Values: Use [] with index.
4. Syntax Rules:
  - Brackets start and end the array.
  - Values separated by commas.
  - Can span multiple lines.
5. Arrays can hold any value, including arrays.
6. typeof operator on Array Returns Object.

# 49. Array Properties & Methods

1. `Array.isArray()` checks if a variable is an array.
2. `Length` property holds the size of the array.
3. Common Methods:
  - `push/pop`: Add or remove to end.
  - `shift/unshift`: Add or remove from front.
  - `splice`: Add or remove elements.
  - `toString`: Convert to string.
  - `sort`: Sort elements.
  - `valueOf`: Get array itself.
4. Arrays also use `reference` like objects.
5. De-structuring also works for Arrays.

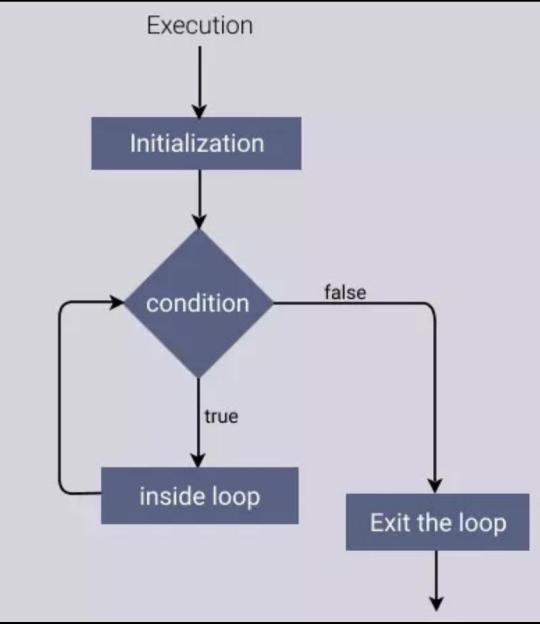


# 50. What is a Loop?



1. Code that runs multiple times based on a condition.
2. Loops also alter the flow of execution, similar to functions.
  - Functions: Reusable blocks of code.
  - Loops: Repeated execution of code.
3. Loops automate repetitive tasks.
4. Types of Loops: for, while, do-while.
5. Iterations: Number of times the loop runs.

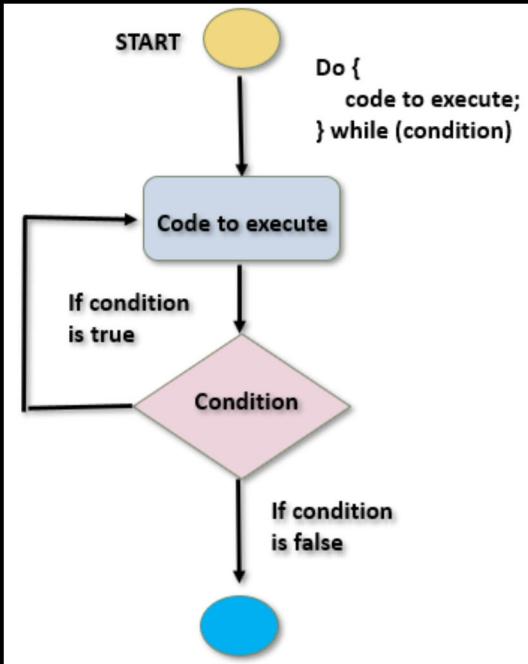
# 51. While Loop



```
while (condition) {  
    // Body of the loop  
}
```

1. Iterations: Number of **times** the loop **runs**.
2. Used for **non-standard** conditions.
3. Repeating a block of code **while** a condition is **true**.
4. Remember: Always include an update to **avoid infinite loops**.

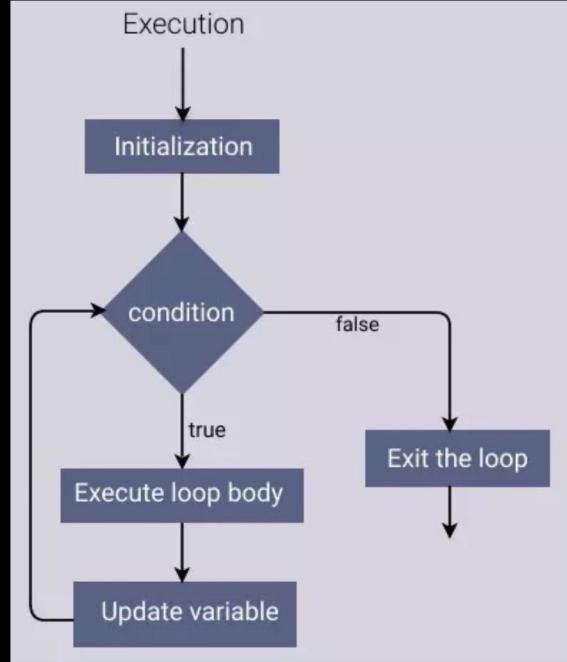
# 52. Do While Loop



```
do {  
    // Body of the loop  
}  
while (condition);
```

1. Executes block first, then checks condition.
2. Guaranteed to run at least one iteration.
3. Unlike while, first iteration is unconditional.
4. Don't forget to update condition to avoid infinite loops.

# 53. For Loop



```
for (initialisation; condition; update) {  
    // Body of the loop  
}
```

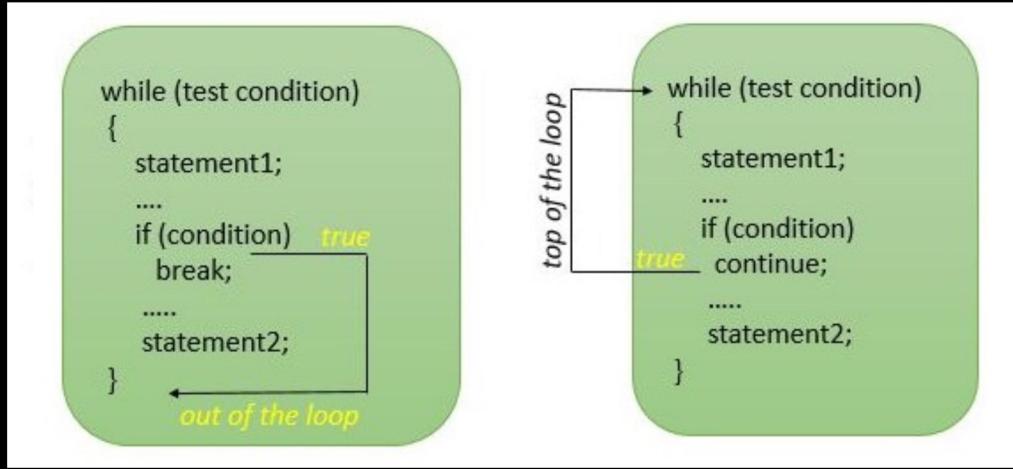
1. Standard loop for running code multiple times.
2. Generally preferred for counting iterations.

# 54. Accumulator Pattern



1. A pattern to accumulate values through looping.
2. Common Scenarios:
  - Sum all the numbers in an array.
  - Create a modified copy of an array.

# 55. Break & Continue

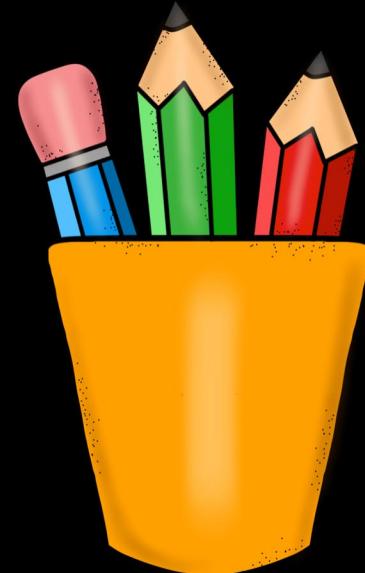


1. **Break** lets you stop a **loop** early, or **break out** of a loop
2. **Continue** is used to skip one iteration or the current iteration
3. In **while loop** remember to do the **increment** manually before using **continue**

# Arrays & Loops

## Revision

47. What is an Array?
48. Array Syntax & Values
49. Array Properties & Methods
50. What is a Loop?
51. While Loop
52. Do While Loop
53. For Loop
54. Accumulator Pattern
55. Break & Continue



# Practice Exercise

## Arrays & Loops

1. Create an array of numbers [5,6]. Add 4 at the beginning and 7 at the end of the array.
2. Create a method to return an element at a particular position in the array.
3. Create an array copy using slice method.
4. Create a while loop that exits after counting 5 prime numbers.
5. Modify the above loop to finish using break.
6. Create a for loop that prints number 1 to 10 in reverse order.
7. Using continue only print positive numbers from the given array [1,-6,5,7,-98]
8. Using accumulator pattern concatenate all the strings in the given array ['KG', 'Coding', 'Javascript', 'Course', 'is', 'Best']



# KG Coding

Some Other One shot Video Links:

- [Complete HTML](#)
- [Complete CSS](#)
- [Complete JavaScript](#)
- [Complete React and Redux](#)
- [One shot University Exam Series](#)



<http://www.kgcoding.in/>

Our YouTube Channels

[KG Coding Android App](#)



[KG Coding](#)



[Knowledge GATE](#)



[KG Placement Prep](#)



[Sanchit Socket](#)

# Advance Functions

- 56. Anonymous Functions As Values
- 57. Arrow functions
- 58. setTimeout & setInterval
- 59. Event Listener
- 60. For Each Loop
- 61. Array methods

# 56. Anonymous Functions As Values

```
let sum = function(num1, num2) {  
    return num1 + num2;  
}
```

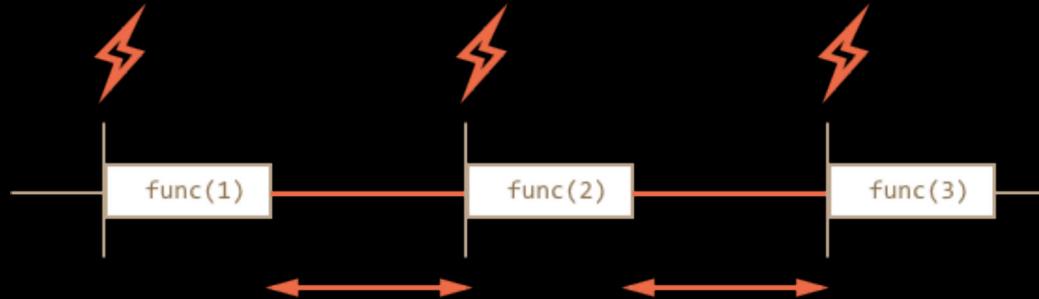
1. Functions in JavaScript are **first-class citizens**; they can be assigned to variables.
2. Functions defined **without a name**, often assigned to a variable.
3. Anonymous functions can be **properties** in objects
4. Can be passed as arguments to other functions.
5. Invoked using **()** after the function **name** or **variable**.
6. **console.log(myFunction);** and **typeof myFunction** will both indicate **it's a function**.

# 57. Arrow Functions

```
let sum = function(num1, num2) {  
    return num1 + num2;  
}  
  
let Sum1 = (num1, num2) => {  
    return num1 + num2;  
}  
  
let Sum2 = (num1, num2) => num1 + num2;  
let square = num => num * num;
```

1. A concise way to write **anonymous** functions.
2. For Single Argument: **Round brackets** optional.
3. For Single Line: **Curly brackets** and return optional.
4. Often used when passing **functions as arguments**.

# 58. setTimeout & setInterval



1. Functions for executing code **asynchronously** after a delay.
2. `setTimeout` runs **once**; `setInterval` runs **repeatedly**
3. **setTimeout**:
  - Syntax: `setTimeout(function, time)`
  - Cancel: `clearTimeout(timerID)`
4. **setInterval**:
  - Syntax: `setInterval(function, time)`
  - Cancel: `clearInterval(intervalID)`

# 59. Event Listener

```
const button = document.querySelector(".btn")  
  
button.addEventListener("click", event => {  
  console.log("Hello!");  
})
```

1. What Is an Event: Occurrences like **clicks, mouse movement, keyboard input** (e.g., birthday, functions).
2. Using **querySelector** to attach listeners.
3. Multiple Listeners: You can add **more than one**.
4. **removeEventListener('event', functionVariable);**

# 60. For Each Loop

```
let foods = ['bread', 'rice', 'meat', 'pizza'];

foods.forEach(function(food) {
    console.log(food);
})
```

1. A method for array iteration, often preferred for readability.
2. Parameters: One for item, optional second for index.
3. Using return is similar to continue in traditional loops.
4. Not straightforward to break out of a forEach loop.
5. When you need to perform an action on each array element and don't need to break early.

# 61. Array Methods

```
[🍔, 🍠, 🐔, 🌶].map(cook) => [🍔, 🍟, 🍗, 🍷]  
[🍔, 🍟, 🍗, 🍷].filter(isVegetarian) => [🍟, 🍷]
```

## 1. Filter Method:

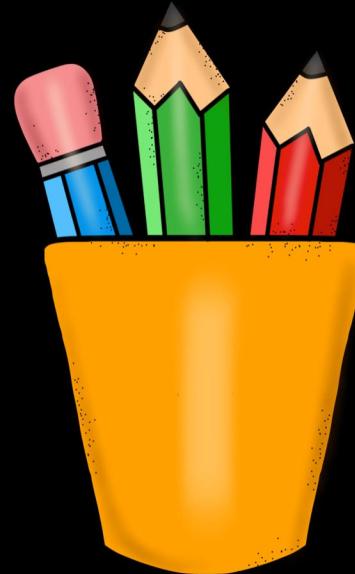
- Syntax: `array.filter((value, index) => return true/false)`
- Use: **Filters** elements based on **condition**.

## 2. Map Method:

- Syntax: `array.map((value) => return newValue)`
- Use: **Transforms** each element.

# Advance Functions Revision

- 56. Anonymous Functions As Values
- 57. Arrow functions
- 58. setTimeout & setInterval
- 59. Event Listener
- 60. For Each Loop
- 61. Array methods



# Practice Exercise

## Advance Functions

1. Create a variable **multiply** and assign a **function** to this variable that **multiplies two numbers**. Call this method from the variable.
2. Create a function **runTwice** that takes a function as a parameter and then **runs that method** twice.
3. Create a button which should **grow** double in size on clicking after **2 seconds**.
4. In the **above exercise** add the behavior using an **event listener** instead of **onclick**.
5. Create a function that **sums** an array of numbers. Do this using a **for-each** loop.
6. Create a function that takes an **array of numbers** and return their **squares** using **map** function.



# KG Coding

Some Other One shot Video Links:

- [Complete HTML](#)
- [Complete CSS](#)
- [Complete JavaScript](#)
- [Complete React and Redux](#)
- [One shot University Exam Series](#)



<http://www.kgcoding.in/>

Our YouTube Channels

[KG Coding Android App](#)



[KG Coding](#)



[Knowledge GATE](#)



[KG Placement Prep](#)



[Sanchit Socket](#)