**Handlebars**

**Definition**

Handlebars.js is a lightweight templating engine that helps in dynamically generating HTML content by separating data (logic) from the presentation (UI). It uses a simple syntax with placeholders ({{ }}) to embed dynamic data into HTML templates.

The main goal of Handlebars is to keep templates clean and readable while making it easy to insert dynamic content into web pages.

Handlebars is used to:

* Simplify rendering of dynamic content in web pages.
* Maintain cleaner and more readable code by separating logic and HTML.
* Improve code reusability through partials (small reusable templates).
* Enhance maintainability by keeping templates and data independent.

**Features of Handlebars**

* **Expressions ({{}})**: Inserts dynamic values into HTML.
* **Helpers**: Functions that process data before rendering.
* **Partials**: Reusable template components.
* **Conditionals ({{#if}}, {{#else}})**: Adds logic to templates.
* **Loops ({{#each}})**: Iterates over lists of data.

**Handlebars Templates**

**Definition:**

A Handlebars template is an HTML structure containing placeholders ({{}}) for dynamic content.

**Purpose:**

* Reduces repetitive code.
* Allows easy modification of UI without changing JavaScript logic.

**Usage:**

Used to dynamically generate HTML with data.

**Syntax**:

<script id="my-template" type="text/x-handlebars-template">

<p>Welcome, {{user}}!</p>

</script>

**Expressions ({{}})**

**Definition:**

Expressions are placeholders in Handlebars that are replaced by actual values when rendering a template.

**Purpose:**

* To insert dynamic data inside an HTML template.

**Usage:**

Used within a template to display variables.

**Syntax:**

<p>{{variableName}}</p>

**Helpers**

**Definition**

Helpers are functions that allow adding custom logic in Handlebars templates.

**Purpose**

* To perform operations like comparisons, loops, or transformations.
* To make templates more flexible and powerful.

**Usage**

* Built-in helpers (if, each, unless, etc.).
* Custom helpers.

**Conditionals ({{#if}} ... {{/if}})**

**Definition:**

Conditional statements allow you to show or hide content based on a condition.

**Purpose:**

* Helps in displaying content only when certain conditions are met.

**Usage:**

Used when you need to check conditions before rendering elements.

**Syntax:**

{{#if condition}}

<p>Condition is true!</p>

{{/if}}

**Loops ({{#each}})**

**Definition:**

Loops in Handlebars allow you to iterate over an array of objects and generate repetitive HTML elements dynamically.

**Purpose:**

* Helps in rendering lists, tables, or repeated components dynamically.
* Reduces redundant code by iterating over data structures.

**Usage:**

Used when you need to display multiple similar items from an array.

**Syntax:**

{{#each arrayName}}

<p>{{this}}</p>

{{/each}}

**Using Objects Inside Loops**

**Definition:**

When iterating over objects inside an array, you can access properties of each object.

**Purpose:**

* Helps display structured data dynamically.
* Commonly used for tables, lists, or detailed views.

**Usage:**

Used when each item in an array has multiple properties.

**Syntax:**

{{#each arrayName}}

<p>Name: {{name}}, Price: {{price}}</p>

{{/each}}

**unless Helper**

**Definition**

The unless helper is the opposite of if. It renders content only if the condition is false.

**Purpose**

* To handle negative conditions without using if and ! (not operator).
* **Syntax:**

{{#unless condition}}

Content if condition is false.

{{/unless}}

**Basic Blocks**

**Definition**

A basic block helper in Handlebars wraps around a section of a template and allows custom logic to be applied to that section.

**Purpose**

* To create reusable template logic.
* To apply transformations or conditional rendering inside the block.

**Usage**

* Define a block helper using Handlebars.registerHelper().
* Use {{#helperName}}...{{/helperName}} to wrap content.

**Syntax**

Handlebars.registerHelper("shout", function(options) {

return options.fn(this).toUpperCase();

});

**Basic Block Variations**

**Definition**

A block helper can provide alternate content when a condition is not met using else.

**Purpose**

* To handle cases where the primary block does not render.
* To provide fallback content.

**Usage**

* Use {{else}} inside a block helper.

**Syntax**

{{#if isIndian}}

<p>Namaste! Welcome to India.</p>

{{else}}

<p>Welcome, dear visitor!</p>

{{/if}}

**The with Helper**

**Definition**

The with helper allows narrowing down the context of a template, making it easier to access properties of an object.

**Purpose**

* To avoid repetitive object references.
* To simplify nested data handling.

**Usage**

* Use {{#with object}} ... {{/with}} to change the context.

**Syntax**

{{#with guest}}

<p>Name: {{name}}</p>

<p>Room No: {{room}}</p>

{{/with}}

**Simple Iterators (each Helper)**

**Definition**

The each helper loops over an array and renders content for each item.

**Purpose**

* To dynamically generate lists.

**Usage**

* Use {{#each array}} ... {{/each}}.

**Syntax**

<ul>

{{#each students}}

<li>{{this}}</li>

{{/each}}

</ul>

**Conditionals (if, unless)**

**Definition**

Conditional helpers render content based on boolean values.

**Purpose**

* To handle logic inside templates.

**Usage**

* Use {{#if condition}} ... {{/if}}.
* Use {{#unless condition}} ... {{/unless}}.

**Syntax**

{{#if isFestival}}

<p>Happy Diwali!</p>

{{else}}

<p>Have a great day!</p>

{{/if}}

**Hash Arguments**

**Definition**

Hash arguments allow passing extra options to helpers.

**Purpose**

* To customize helper behavior.

**Usage**

* Use key=value inside helper calls.

**Syntax**

{{greet name="Amit" greeting="Namaste"}}

**Block Parameters**

**Definition**

Block parameters allow passing context values into a block.

**Purpose**

* To make templates more flexible.

**Usage**

* Use as |variable| inside helpers.

**Syntax**

{{#each students as |student|}}

<p>{{student}}</p>

{{/each}}

**Raw Blocks**

**Definition**

Raw blocks allow content to remain untouched.

**Usage**

* Use {{{{raw}}}} ... {{{{/raw}}}}.

**Syntax**

{{{{raw}}}}

{{name}} will not be processed.

{{{{/raw}}}}