FurniCraft Website Documentation

Combined HTML, JavaScript, and CSS Documentation

HTML Documentation for FurniCraft Website

Table of Contents

Basic Structure

```
<!DOCTYPE html>
<html lang="en" class="scroll-smooth">
<head>
    <!-- Meta tags and resources -->
</head>
<body class="bg-gray-50">
    <!-- Website content -->
</body>
</html>
```

- `<!DOCTYPE html>`: Declares the document type as HTML5
- `lang="en"`: Specifies the language as English
- `class="scroll-smooth"`: Enables smooth scrolling behavior
- `class="bg-gray-50"`: Sets a light gray background color using Tailwind CSS

Meta Tags

```
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>FurniCraft - Premium Furniture Store</title>
```

- `charset="UTF-8"`: Specifies character encoding
- `viewport`: Makes the website responsive on mobile devices
- `title`: Sets the browser tab title

External Resources

```
<script src="https://cdn.tailwindcss.com"></script>
<script
src="https://cdnjs.cloudflare.com/ajax/libs/flowbite/2.2.1/flowbite.min.js">
</script>
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/6.5.1/css/all.min.css">
<link rel="stylesheet" href="styles.css">
```

- Tailwind CSS: Utility-first CSS framework
- Flowbite: UI component library
- Font Awesome: Icon library
- Custom CSS file

Navigation Bar

```
<nav class="bg-white shadow-lg fixed w-full z-[9999] top-0 left-0">
    <div class="max-w-7xl mx-auto px-4">
        <div class="flex justify-between items-center h-16">
            <!-- Logo -->
            <div class="flex items-center">
                <a href="#" class="text-2xl font-bold text-indigo-
600">FurniCraft</a>
            </div>
            <!-- Desktop Menu -->
            <div class="hidden md:flex items-center space-x-8">
                <!-- Navigation Links -->
            </div>
            <!-- Mobile Menu Button -->
            <div class="md:hidden">
                <button class="mobile-menu-button">
                    <i class="fas fa-bars w-6 h-6"></i>
                </button>
            </div>
        </div>
    </div>
</nav>
```

- `fixed`: Makes the navigation bar stick to the top
- `z-[9999]`: Ensures the nav stays above other elements
- `hidden md:flex`: Hides on mobile, shows on medium screens
- `md:hidden`: Shows on mobile, hides on medium screens

Hero Section

```
<div id="home" class="relative h-screen pt-16">
    <div id="default-carousel" class="relative h-full" data-carousel="slide">
        <!-- Carousel Items -->
    </div>
</div>
```

- `h-screen`: Sets height to 100% of viewport height
- `pt-16`: Adds padding-top to account for fixed navbar
- `data-carousel`: Flowbite carousel attributes

Products Section

```
<section id="products" class="max-w-7xl mx-auto px-4 py-12">
    <h2 class="text-3xl font-bold text-center mb-8">Featured Furniture</h2>
    <div class="products-grid grid grid-cols-1 sm:grid-cols-2 md:grid-cols-3
lg:grid-cols-4 gap-6">
```

```
            <!-- Products will be dynamically inserted here -->
        </div>
    </section>
```

- `max-w-7xl`: Sets maximum width
- `grid-cols-1 sm:grid-cols-2`: Responsive grid layout
- `gap-6`: Adds spacing between grid items

Cart Section

```
<section id="cart" class="max-w-7xl mx-auto px-4 py-12 bg-gray-100">
    <h2 class="text-3xl font-bold text-center mb-8">Your Cart</h2>
    <div class="cart-items space-y-4">
        <!-- Cart items will be dynamically inserted here -->
    </div>
    <div class="cart-summary mt-8 p-6 bg-white rounded-lg shadow-md">
        <!-- Cart summary details -->
    </div>
</section>
```

- `space-y-4`: Adds vertical spacing between cart items
- `rounded-lg`: Rounds the corners
- `shadow-md`: Adds medium shadow effect

Contact Section

```
<section id="contact" class="max-w-7xl mx-auto px-4 py-12 bg-gray-100">
    <h2 class="text-3xl font-bold text-center mb-8">Contact Us</h2>
    <div class="max-w-lg mx-auto">
        <form class="space-y-4">
            <!-- Form fields -->
        </form>
    </div>
</section>
```

- `max-w-lg`: Sets maximum width for the form
- `space-y-4`: Adds vertical spacing between form elements

Footer

```
<footer class="bg-gray-800 text-white py-8">
    <div class="max-w-7xl mx-auto px-4">
        <div class="grid grid-cols-1 md:grid-cols-3 gap-8">
            <!-- Footer content -->
        </div>
    </div>
</footer>
```

- `bg-gray-800`: Dark background color
- `grid-cols-1 md:grid-cols-3`: Responsive grid layout
- `gap-8`: Adds spacing between grid items

Common Tailwind CSS Classes Used

- `flex`: Creates a flex container
- `items-center`: Centers items vertically
- `justify-between`: Spaces items evenly
- `text-center`: Centers text
- `rounded`: Rounds corners
- `shadow`: Adds shadow effect
- `hover:`: Applies styles on hover
- `transition`: Adds smooth transitions
- `bg-{color}-{shade}`: Background colors
- `text-{color}-{shade}`: Text colors
- `p-{size}`: Padding
- `m-{size}`: Margin
- `w-{size}`: Width
- `h-{size}`: Height

JavaScript Documentation for FurniCraft Website

Table of Contents

Global Variables

```
let cart = [];
const DELIVERY_CHARGE = 50;
const SHIPPING_CHARGE = 100;
const DISCOUNT_THRESHOLD = 1000;
const DISCOUNT_PERCENTAGE = 10;
const FREE_DELIVERY_THRESHOLD = 1500;
```

- `cart`: Array to store cart items
- Constants for various charges and thresholds
- Purpose: Centralized configuration and state management

DOM Elements

```
const productsGrid = document.querySelector('.products-grid');
const cartItems = document.querySelector('.cart-items');
const subtotalElement = document.querySelector('.subtotal');
// ... more elements
```

- `document.querySelector()`: Selects first matching element
- Purpose: Caches DOM elements for better performance

Initialization

```
document.addEventListener('DOMContentLoaded', () => {
    fetchProducts();
    initializeReviews();
    initializeMobileMenu();
    updateCartSummary();
});
```

- `DOMContentLoaded`: Event fires when HTML is fully loaded
- Purpose: Initializes all website features

Product Management

Fetching Products

```
async function fetchProducts() {
    const response = await fetch('https://alchemist-kira.github.io/shop-
api/product.json');
    const data = await response.json();
    displayProducts(data.products);
}
```

- `async/await`: Handles asynchronous operations
- `fetch`: Gets data from API
- Purpose: Loads product data from external source

Displaying Products

```
function displayProducts(products) {
    productsGrid.innerHTML = products.map(product => `
        <div class="product-card">
            <!-- Product HTML template -->
        </div>
    `).join('');
}
```

- Template literals: Creates HTML strings
- `map()`: Transforms array of products to HTML
- `join('')`: Combines array into string
- Purpose: Renders products in the grid

Cart Management

Adding to Cart

```
function addToCart(product) {
    const existingItem = cart.find(item => item.id === product.id);

    if (existingItem) {
        existingItem.quantity += 1;
    } else {
        cart.push({
            ...product,
            quantity: 1
        });
    }

    updateCart();
    updateCartSummary();
}
```

- `find()`: Searches array for matching item
- Spread operator: Copies object properties
- Purpose: Manages cart item addition

Updating Cart

```
function updateCart() {
    cartItems.innerHTML = cart.map(item => `
        <div class="cart-item">
            <!-- Cart item HTML template -->
        </div>
```

```
    `).join('');
}
```

- Purpose: Refreshes cart display

Cart Summary

```
function updateCartSummary() {
    if (!subtotalElement || !deliveryChargeElement || !shippingChargeElement ||
!discountElement || !totalElement) return;

    const subtotal = cart.reduce((total, item) => total + (item.price *
item.quantity), 0);
    const discount = subtotal >= DISCOUNT_THRESHOLD ? (subtotal *
DISCOUNT_PERCENTAGE / 100) : 0;

    // Apply free delivery if subtotal is over threshold
    const finalDeliveryCharge = cart.length > 0 ? (subtotal >=
FREE_DELIVERY_THRESHOLD ? 0 : DELIVERY_CHARGE) : 0;
    const finalShippingCharge = cart.length > 0 ? (subtotal >=
FREE_DELIVERY_THRESHOLD ? 0 : SHIPPING_CHARGE) : 0;

    const total = subtotal + finalDeliveryCharge + finalShippingCharge - discount;

    subtotalElement.textContent = `$${subtotal.toFixed(2)}`;
    deliveryChargeElement.textContent = cart.length > 0 ?
`$${finalDeliveryCharge.toFixed(2)}` : '$0.00';
    shippingChargeElement.textContent = cart.length > 0 ?
`$${finalShippingCharge.toFixed(2)}` : '$0.00';
    discountElement.textContent = `$${discount.toFixed(2)}`;
    totalElement.textContent = `$${total.toFixed(2)}`;
}
```

How Cart Summary Works

1. **Initial Check**

```
if (!subtotalElement || !deliveryChargeElement || !shippingChargeElement ||
!discountElement || !totalElement) return;
```

   - Checks if all required DOM elements exist
   - Returns early if any element is missing to prevent errors

2. **Subtotal Calculation**

```
const subtotal = cart.reduce((total, item) => total + (item.price *
item.quantity), 0);
```

   - Uses `reduce()` to sum up all items in cart
   - For each item: price × quantity
   - Starts from 0 (initial value)

3. **Discount Calculation**

```
const discount = subtotal >= DISCOUNT_THRESHOLD ? (subtotal *
DISCOUNT_PERCENTAGE / 100) : 0;
```

- Checks if subtotal meets discount threshold ($1000)
- If yes: applies 10% discount
- If no: no discount (0)

4. **Delivery and Shipping Charges**

```
const finalDeliveryCharge = cart.length > 0 ? (subtotal >=
FREE_DELIVERY_THRESHOLD ? 0 : DELIVERY_CHARGE) : 0;
const finalShippingCharge = cart.length > 0 ? (subtotal >=
FREE_DELIVERY_THRESHOLD ? 0 : SHIPPING_CHARGE) : 0;
```

- First checks if cart has items (`cart.length > 0`)
- If cart is empty: charges are 0
- If cart has items:
  - Checks if subtotal meets free delivery threshold ($1500)
  - If yes: charges are 0
  - If no: applies normal charges ($50 delivery, $100 shipping)

5. **Total Calculation**

```
const total = subtotal + finalDeliveryCharge + finalShippingCharge -
discount;
```

- Adds all components: subtotal + delivery + shipping
- Subtracts discount
- This gives final amount to pay

6. **Display Updates**

```
subtotalElement.textContent = `$${subtotal.toFixed(2)}`;
deliveryChargeElement.textContent = cart.length > 0 ?
`$${finalDeliveryCharge.toFixed(2)}` : '$0.00';
shippingChargeElement.textContent = cart.length > 0 ?
`$${finalShippingCharge.toFixed(2)}` : '$0.00';
discountElement.textContent = `$${discount.toFixed(2)}`;
totalElement.textContent = `$${total.toFixed(2)}`;
```

   - Updates each element with calculated values
   - Uses `toFixed(2)` to show 2 decimal places
   - Shows $0.00 for charges when cart is empty
   - Formats all numbers as currency with $ symbol

Example Scenarios

1. **Empty Cart**
   - Subtotal: $0.00
   - Delivery: $0.00
   - Shipping: $0.00
   - Discount: $0.00
   - Total: $0.00
2. **Cart with $800 Total**
   - Subtotal: $800.00
   - Delivery: $50.00
   - Shipping: $100.00
   - Discount: $0.00 (below $1000 threshold)
   - Total: $950.00
3. **Cart with $1200 Total**
   - Subtotal: $1200.00
   - Delivery: $50.00
   - Shipping: $100.00
   - Discount: $120.00 (10% of $1200)
   - Total: $1230.00
4. **Cart with $1600 Total**
   - Subtotal: $1600.00
   - Delivery: $0.00 (free delivery over $1500)
   - Shipping: $0.00 (free shipping over $1500)
   - Discount: $160.00 (10% of $1600)
   - Total: $1440.00

Placing Order

```
function placeOrder() {
    if (cart.length === 0) return;

    const confirmationMessage = document.createElement('div');
    confirmationMessage.className = 'bg-green-100 border border-green-400 text-
green-700 px-4 py-3 rounded relative mb-4';
    // ... message content

    cart = [];
    updateCart();
    updateCartSummary();
}
```

- `createElement()`: Creates new DOM element
- Purpose: Handles order placement and confirmation

Review System

```
const reviews = [
    {
        name: "Mahfuz",
        rating: 5,
        comment: "Amazing products and fast delivery!"
    },
    // ... more reviews
];

function initializeReviews() {
    const reviewsContainer = document.querySelector('#reviews .grid');
    reviewsContainer.innerHTML = reviews.map(review => `
        <!-- Review HTML template -->
    `).join('');
}
```

- Array of review objects
- Purpose: Displays customer reviews

Mobile Menu

```
function initializeMobileMenu() {
    mobileMenuButton.addEventListener('click', () => {
        mobileMenu.classList.toggle('hidden');
    });
}
```

- `addEventListener()`: Attaches event handler
- `classList.toggle()`: Toggles CSS class
- Purpose: Handles mobile menu functionality

JavaScript Concepts Used

Variables and Constants

- `let`: Mutable variables
- `const`: Immutable variables
- Purpose: Data storage and configuration

Functions

- Regular functions
- Arrow functions
- Async functions
- Purpose: Code organization and reusability

Array Methods

- `map()`: Transform array elements
- `find()`: Search array
- `reduce()`: Calculate totals
- `filter()`: Filter array elements
- Purpose: Data manipulation

DOM Manipulation

- `querySelector()`: Select elements
- `innerHTML`: Update content
- `createElement()`: Create elements
- `classList`: Manage classes
- Purpose: Update webpage content

Events

- `addEventListener()`: Handle events
- `DOMContentLoaded`: Page load
- `click`: User clicks
- Purpose: User interaction handling

Async/Await

- `async`: Declare async function
- `await`: Wait for promise
- `fetch()`: Get data
- Purpose: Handle asynchronous operations

Template Literals

- Backticks (`)
- `${expression}`
- Purpose: Create dynamic strings

Object Methods

- Spread operator
- Object destructuring
- Purpose: Object manipulation

Error Handling

- Try/catch blocks
- Error messages
- Purpose: Graceful error management

CSS Documentation for FurniCraft Website

Table of Contents

Product Card Hover Effect

```
.product-card {
    transition: transform 0.3s ease-in-out;
}

.product-card:hover {
    transform: translateY(-5px);
}
```

- `transition`: Creates smooth animation for transform property
- `transform`: Moves element up by 5 pixels on hover
- `ease-in-out`: Smooth acceleration and deceleration
- Purpose: Creates an interactive "lifting" effect when hovering over products

Scrollbar Styling

```
::-webkit-scrollbar-thumb:hover {
    background: #555;
}
```

- `::-webkit-scrollbar-thumb`: Targets the scrollbar thumb (draggable part)
- `:hover`: Applies style when mouse hovers over scrollbar
- `background: #555`: Changes scrollbar color to dark gray
- Purpose: Improves scrollbar visibility and user experience

Form Input Focus Styles

```
input:focus, textarea:focus {
    outline: none;
    border-color: #4F46E5;
    box-shadow: 0 0 0 3px rgba(79, 70, 229, 0.1);
}
```

- `:focus`: Applies styles when input is selected
- `outline: none`: Removes default browser outline
- `border-color`: Changes border color to indigo
- `box-shadow`: Adds subtle glow effect
- Purpose: Provides visual feedback when interacting with form elements

Button Hover Effects

```
button {
    transition: all 0.3s ease-in-out;
}

button:hover {
    transform: translateY(-1px);
}
```

- `transition: all`: Animates all changing properties
- `transform`: Slightly lifts button on hover
- Purpose: Makes buttons more interactive and engaging

Cart Item Animation

```
.cart-item {
    animation: slideIn 0.3s ease-in-out;
}

@keyframes slideIn {
    from {
        opacity: 0;
        transform: translateX(-20px);
    }
    to {
        opacity: 1;
        transform: translateX(0);
    }
}
```

- `animation`: Applies the slideIn animation
- `@keyframes`: Defines the animation sequence
- `from/to`: Specifies start and end states
- `opacity`: Controls transparency
- `transform`: Moves element horizontally
- Purpose: Creates smooth entrance animation for cart items

CSS Properties Explained

Transitions

- `transition`: Shorthand for transition properties
- `transition-property`: Which properties to animate
- `transition-duration`: How long the animation takes
- `transition-timing-function`: How the animation progresses
- `transition-delay`: When the animation starts

Transforms

- `transform`: Applies 2D or 3D transformations
- `translateX/Y`: Moves element horizontally/vertically
- `scale`: Changes element size
- `rotate`: Rotates element
- `skew`: Skews element

Animations

- `animation`: Shorthand for animation properties
- `animation-name`: Name of the keyframes
- `animation-duration`: How long animation takes
- `animation-timing-function`: How animation progresses

- `animation-delay`: When animation starts
- `animation-iteration-count`: How many times to repeat
- `animation-direction`: Which direction to play
- `animation-fill-mode`: How to apply styles before/after
- `animation-play-state`: Whether animation is running/paused

Pseudo-classes

- `:hover`: When mouse is over element
- `:focus`: When element is selected
- `:active`: When element is being clicked
- `:visited`: For visited links
- `:first-child`: First element in parent
- `:last-child`: Last element in parent

Box Model

- `margin`: Space outside element
- `padding`: Space inside element
- `border`: Element's border
- `box-shadow`: Adds shadow effect
- `border-radius`: Rounds corners

Colors and Opacity

- `color`: Text color
- `background-color`: Background color
- `opacity`: Element transparency
- `rgba()`: Color with alpha channel
- `hsla()`: Color with hue, saturation, lightness, alpha

Layout

- `display`: How element is displayed
- `position`: How element is positioned
- `z-index`: Stack order
- `float`: Element floating
- `clear`: Clearing floats

```
```