# Semantic vs Non-Semantic HTML

## Overview

- Semantic HTML: Uses HTML tags that clearly describe their meaning in a human- and machine-readable way.
- Non-Semantic HTML: Uses HTML tags that do not convey meaning about their content.

### Semantic HTML

Definition: Semantic HTML elements clearly describe their meaning both to the browser and the developer. These elements are intended to define the structure and content of the web page.

**Examples:**

1. <header>: Represents a container for introductory content or a set of navigational links.
2. <nav>: Represents a section of a page that links to other pages or parts within the page.
3. <section>: Represents a standalone section of a document, which does not have a more specific semantic element to represent it.
4. <article>: Represents a self-contained composition in a document, page, or site, such as a blog post, newspaper article, etc.
5. <aside>: Represents content that is tangentially related to the content around it.
6. <footer>: Represents a footer for its nearest sectioning content or sectioning root element.
7. <main>: Represents the dominant content of the <body> of a document.

**Benefits:**

1. Accessibility: Improves the accessibility of web pages by providing meaningful structure that assistive technologies can use to navigate and interpret the content.
2. SEO: Enhances search engine optimization by providing clear structure and meaning to the content, making it easier for search engines to index and rank.
3. Readability: Improves the readability and maintainability of the code by providing clear and descriptive tags.
4. Consistency: Promotes a consistent structure and layout across different pages and projects.

Example:

Auto (HTML, XML)

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Semantic HTML Example</title>
</head>
<body>
    <header>
        <h1>Welcome to My Website</h1>
        <nav>
            <ul>
                <li><a href="#home">Home</a></li>
                <li><a href="#about">About</a></li>
                <li><a href="#contact">Contact</a></li>
            </ul>
        </nav>
    </header>
    <main>
        <article>
            <h2>My First Article</h2>
            <p>This is the content of my first article. It's written in
semantic HTML!</p>
        </article>
        <aside>
            <h2>Related Links</h2>
            <ul>
                <li><a href="#link1">Link 1</a></li>
                <li><a href="#link2">Link 2</a></li>
            </ul>
        </aside>
    </main>
    <footer>
        <p>&copy; 2024 My Website</p>
    </footer>
</body>
</html>
```

## Non-Semantic HTML

Definition: Non-semantic HTML elements do not convey any meaning about their content. They are used for layout purposes without providing any indication of what the content within them represents.

**Examples:**

1. <div>: A generic container for flow content that by itself does not represent anything.
2. <span>: A generic inline container for phrasing content that does not convey any meaning.

**Drawbacks:**

1. Accessibility: Does not provide any meaningful structure for assistive technologies, making it harder for users with disabilities to navigate and understand the content.
2. SEO: Less effective for search engine optimization as it does not provide clear structure and meaning to the content.
3. Readability: Makes the code harder to read and maintain, especially for other developers who might work on the same project.

**Example:**

Auto (HTML, XML)

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Non-Semantic HTML Example</title>
</head>
<body>
    <div>
        <h1>Welcome to My Website</h1>
        <div>
            <ul>
                <li><a href="#home">Home</a></li>
                <li><a href="#about">About</a></li>
                <li><a href="#contact">Contact</a></li>
            </ul>
        </div>
    </div>
    <div>
        <div>
            <h2>My First Article</h2>
            <p>This is the content of my first article. It's written in non-semantic HTML!</p>
        </div>
        <div>
            <h2>Related Links</h2>
            <ul>
                <li><a href="#link1">Link 1</a></li>
                <li><a href="#link2">Link 2</a></li>
            </ul>
        </div>
    </div>
    <div>
        <p>&copy; 2024 My Website</p>
    </div>
</body>
</html>
```

### Conclusion

- Semantic HTML is preferred for creating web pages that are accessible, SEO-friendly, and easy to read and maintain.
- Non-Semantic HTML should be used sparingly and mainly for layout purposes where no semantic meaning is required.
- Using semantic elements where appropriate makes the web more understandable for both humans and machines.

# HTML <div> Tag Notes

### Overview

- The <div> tag is a block-level container used to group elements together for organizing content.
- It does not provide any semantic meaning about its content.

## Basic Syntax

Auto

```
<div>Content goes here</div>
```

## Example Usage

## Basic Usage

Auto

```
<div>
    <p>This is a simple div.</p>
</div>
```

## Grouping Elements

Auto (CSS)

```
<div>
  <h2>Article Title</h2>
  <p>Article content goes here...</p>
</div>
<div>
  <h2>Another Article Title</h2>
  <p>More article content...</p>
</div>
```

## Nesting <div> Elements

- <div> tags can be nested to create more complex structures.

Auto (CSS)

```
<div>
  <div>
    <h2>Nested Div 1</h2>
    <p>Content inside the first nested div.</p>
  </div>
  <div>
    <h2>Nested Div 2</h2>
    <p>Content inside the second nested div.</p>
  </div>
</div>
```

## Common Uses

1. Layout: <div> is often used to structure the layout of a webpage, such as creating sections for headers, footers, sidebars, and main content areas.
2. Grouping: Group related content together to form logical sections of the page.
3. Containers: Use <div> to create containers for various elements and content blocks.

## Best Practices

- Use <div> for grouping and layout purposes when no other semantic tag is appropriate.
- Avoid overusing <div> elements to keep the HTML structure clean and readable.
- Use semantic HTML elements like <header>, <footer>, <section>, and <article> when the content has a specific meaning.

## Conclusion

- The <div> tag is a versatile tool for grouping and organizing content on a webpage.
- While it lacks semantic meaning, it is essential for layout and creating containers.
- Use semantic tags where possible, reserving <div> for generic containers and layout structures.

By using these attributes and following best practices, you can effectively utilize the <div> tag to structure and organize your web content.

# HTML <span> Tag Notes

## Overview

- The <span> tag is an inline container used to mark up a part of a text or a part of a document.
- It is used to group inline elements for styling purposes.
- It does not provide any semantic meaning about its content.

## Basic Syntax

Auto

Auto

```
<span>Content goes here</span>
```

## Example Usage

## Basic Usage

Auto

```
<p>This is a <span>simple span</span> example.</p>
```

## Highlighting Text

Auto

```
<p>Here's a <span>highlighted part</span> of this sentence.</p>
```

## Wrapping Text

Auto

```
<p>The quick brown fox jumps over the <span>lazy dog</span>.</p>
```

## Nesting <span> Elements

- <span> tags can be nested to create more complex structures.

Auto

```
<p>This is a <span>nested <span>span</span> example</span>.</p>
```

## Common Uses

1. Styling: Apply styles to a specific part of the text within a larger block of text.
2. Scripting: Use JavaScript to manipulate a specific portion of the text.
3. Inline Grouping: Group inline elements together without affecting the layout.

## Accessibility

- Like <div>, <span> does not provide semantic meaning by itself. It can be combined with ARIA

- Like <div>, <span> does not provide semantic meaning by itself. It can be combined with ARIA attributes to improve accessibility.
- Example with ARIA role:

Auto (HTML, XML)

```
<p>This is a <span role="note">noted part</span> of the sentence.</p>
```

### Best Practices

- Use <span> for grouping and styling inline elements when no other semantic tag is appropriate.
- Avoid overusing <span> elements to keep the HTML structure clean and readable.
- Use semantic HTML elements like <strong>, <em>, <mark>, and <a> when the content has a specific meaning.

### Conclusion

- The <span> tag is a versatile tool for grouping and styling inline content.
- While it lacks semantic meaning, it is essential for applying styles and scripting to specific parts of a text.
- Use semantic tags where possible, reserving <span> for generic inline containers and styling.

## Difference Between <div> and <span>

| | | |
|---|---|---|
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |

|  |  |  |
|---|---|---|

## Conclusion

- <div>: Best used for grouping block-level elements and creating layout structures. It affects the document's structure by creating new blocks on the page.
- <span>: Best used for grouping inline elements and styling parts of a text. It does not affect the document's structure and remains inline with the surrounding content.

By understanding and using these tags appropriately, you can create well-structured, readable, and accessible HTML documents.

inline elements → width

img → replaced inline element

INLINE ELEMENTS → NO WIDTH OR HEIGHT

IMG → SPECIAL KIND OF INLINE ELEMENT → REPLACED INLINE ELEMENT