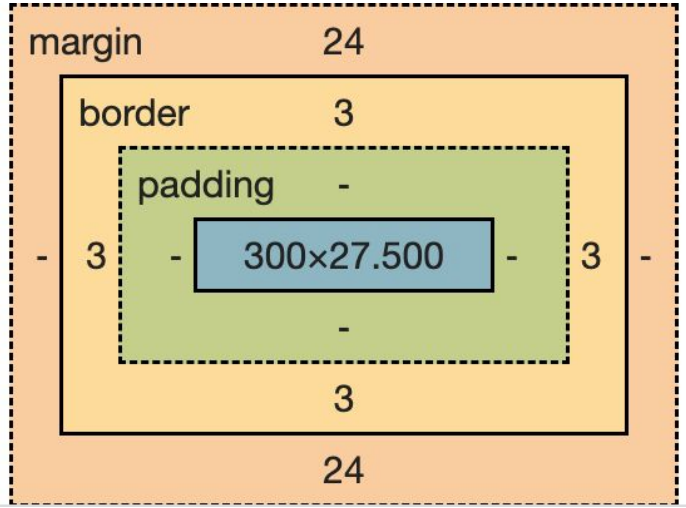
```
<!DOCTYPE html>
<html>
<head>
    <title>CSCI 111 Web Programming and Problem Solving</title>
</head>
<body>
    <h1>Week-4-Lecture</h1>
    <h2>Introduction to CSS Part III</h2>
    <ul>
        <li Dr. Talgat Manglayev</li>
        <li Dr. Irina Dolzhikova</li>
        <li Marat Isteleyev</li>
    </ul>
</body>
</html>
```

outline

```
<ol>
    <li>Box Model</li>
    <ol>
        <li>Box anatomy</li>
        <li>Overflow</li>
        <li>Box sizing</li>
    </ol>
    <li>Positioning the Elements</li>
    <ol>
        <li>Display</li>
        <li>Float</li>
        <li>Position</li>
    </ol>
    <li>Website Layout</li>
    <ol>
        <li>Multi column pages</li>
    </ol>
</ol>
```

Box Model

The CSS **Box Model** is a box that wraps around every HTML element.



Components of Box Model:

- **Content** - actual content of the box.
- **Padding** – transparent area around the content.
- **Border** – a border around the padding and content
- **Margin** – transparent rea outside the border.

Box Model

Some properties of Box Model:

- For margin, padding and border, we can define values separately for the properties such as top, right, bottom and left or use the shorthand notations
  - **margin-top**: 20px;
  - **margin**: 50px 30px 50px 30px;
- If two elements have facing margins, the **maximum** of two gets applied
- If the content is not fitting the box, use **overflow** property
  - **auto** value adds a scrolling
  - **hidden** value hides the extra part of content

Box Sizing

The **width** and **height** properties depend on **box-sizing** property:

- **content-box** includes ***only content*** and not borders and padding
- **border-box** includes ***borders, padding and content*** (recommended way)
- By default, **box-sizing** is **not** inherited by children
- Use the **universal** selector * to set **box-sizing** for each element

```
*{
    box-sizing: border-box;
    margin: 0;
    padding: 0;
}
```

Applied to all elements

Positioning the Elements

Normal Document Flow defines the positions of elements on the screen depending on their place in HTML document.

We can change this flow using positioning properties such as:

- display
- flow
- position

Display

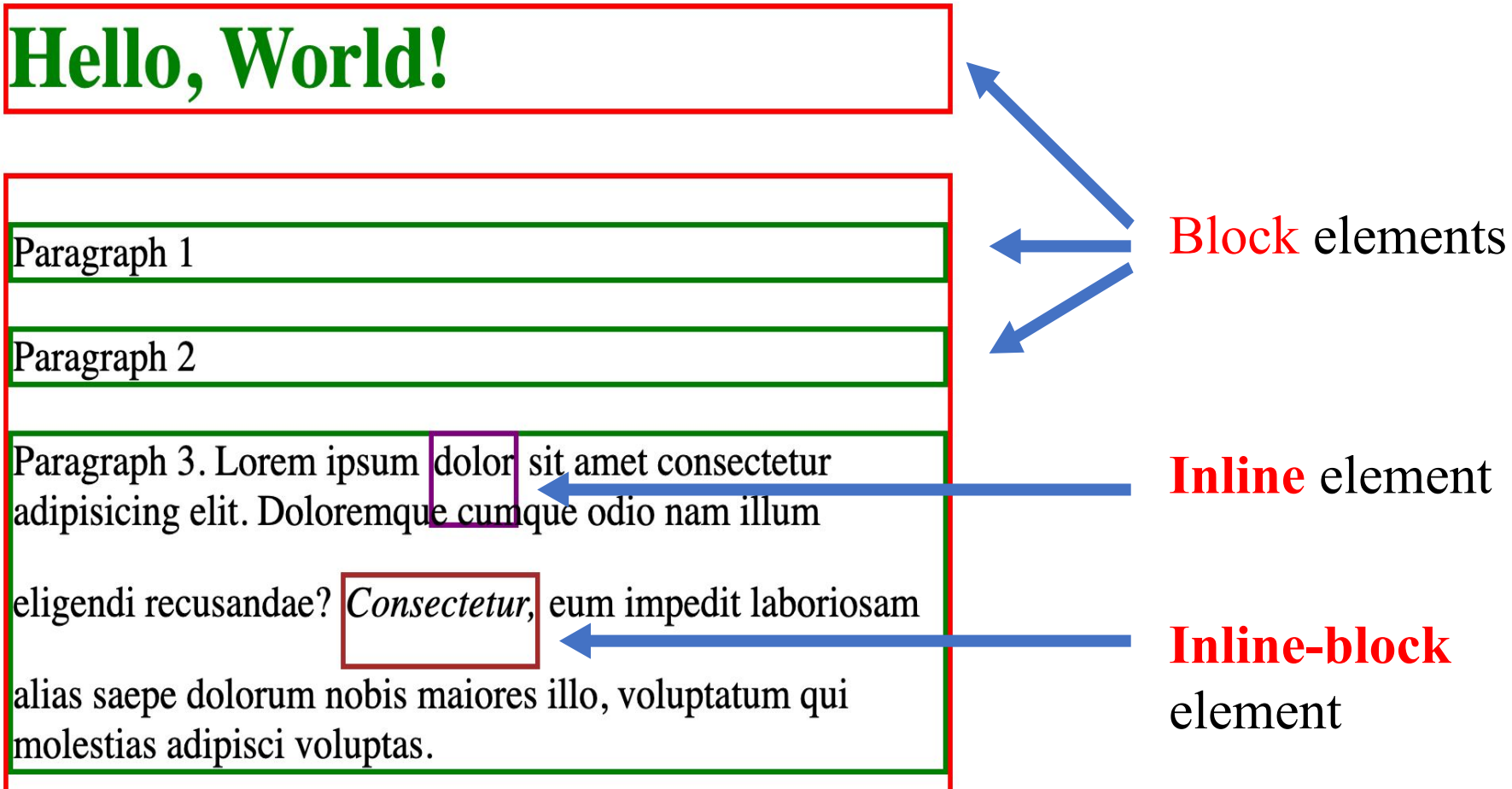
Display property defines how the box model is displayed:

- **block** starts from the new line and occupies it all
- **inline** is placed on the same line where defined

  (some box properties are ignored like **margin-top**)

- **inline-block** behaves like a **block** and **inline** elements
- **flex** makes the elements inside it to line up (**float**)

Display

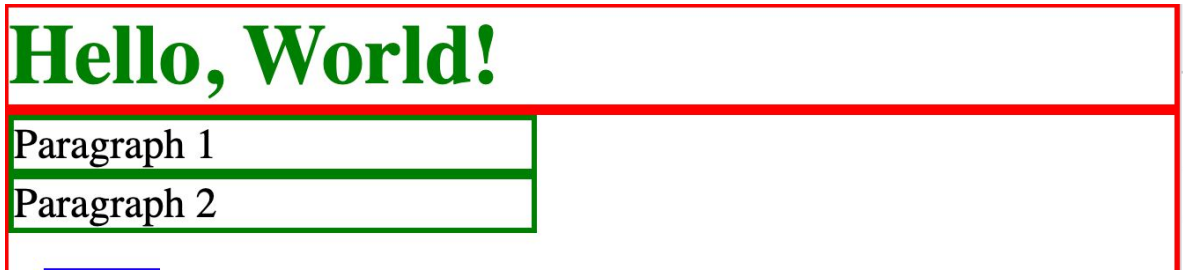

Hello, World!
Paragraph 1
Paragraph 2
Paragraph 3. Lorem ipsum dolor sit amet consectetur adipisicing elit. Doloremque cumque odio nam illum
eligendi recusandae? Consectetur, eum impedit laboriosam
alias saepe dolorum nobis maiores illo, voluptatum qui molestias adipisci voluptas.
Block elements
Inline element
Inline-block element

Float

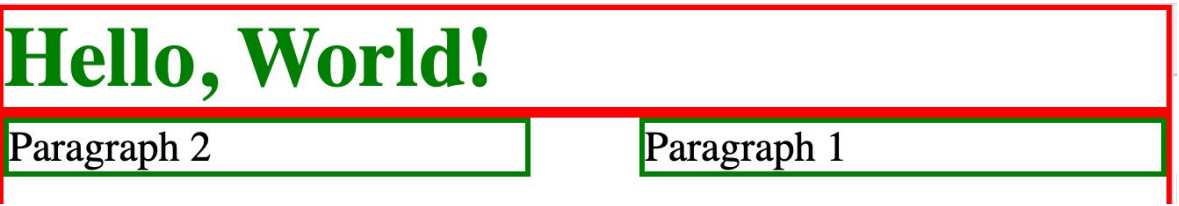
The **float** property specifies how an element should float:

- Elements can float to the right or left (none is default)
- Floated elements are removed from the Normal flow
- The next element occupies the free space of the floated element
- The **clear** property prevents the next element from occupying free space of the floating element

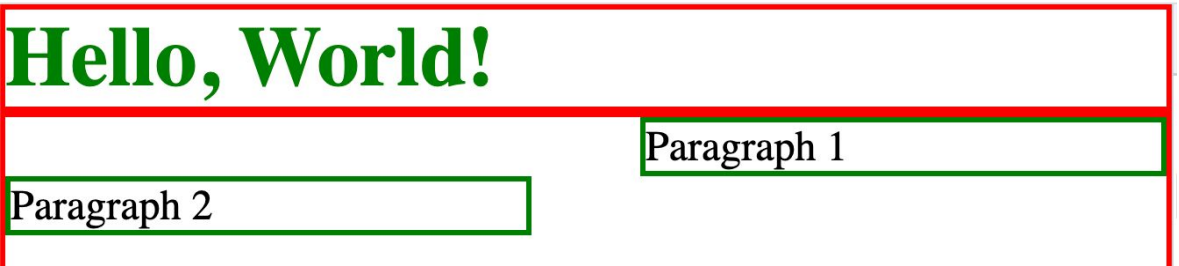
Float



Elements in Normal flow, not floated



Paragraph 1 is floated to the right



Paragraph 1 is floated to the right, Paragraph 2 is **cleared**

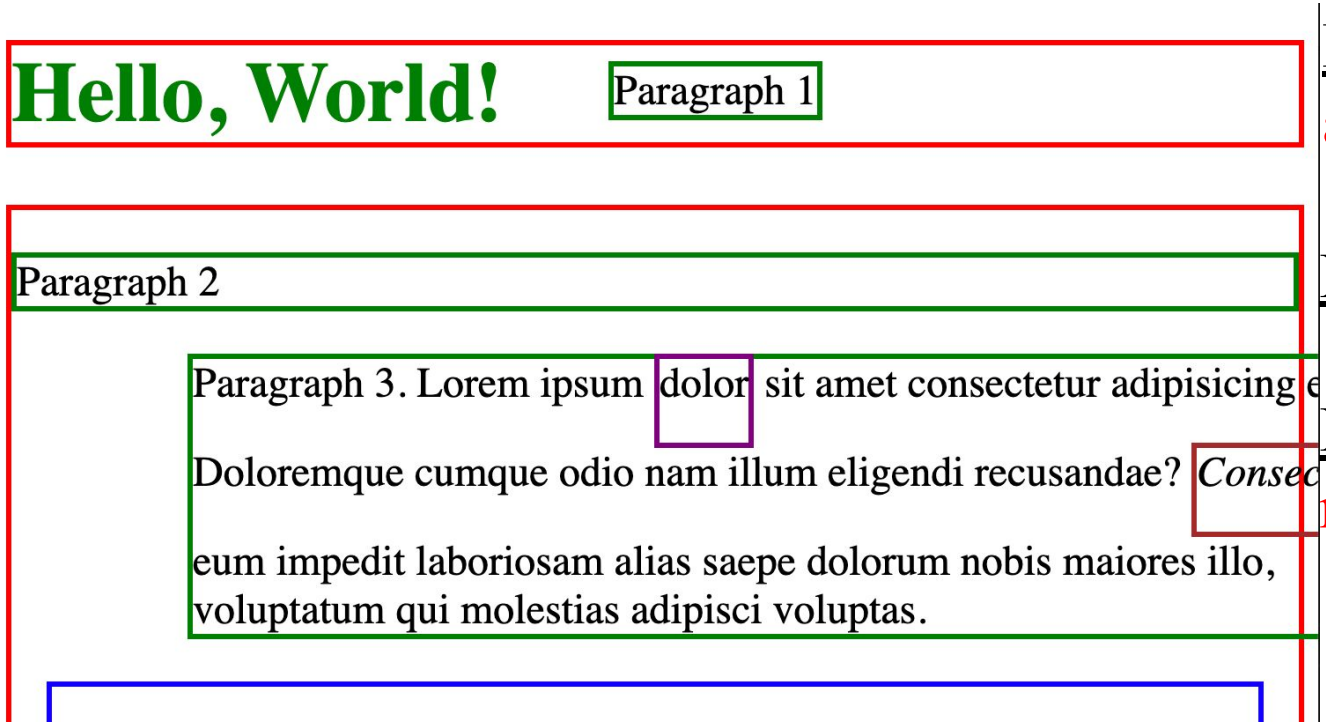
Position

Position property specifies the type of positioning method used for an element:

- **static** - positioned according to the normal flow (default)
- **relative** - positioned relative to its normal position (**preserves** space)
- **absolute** - positioned relative to the nearest positioned ancestor (**removes** space)
- **fixed** - positioned relative to the viewport (**removes** space)
- **sticky** - positioned based on the user's scroll position (as relative or fixed)

- Elements are positioned using the **top, bottom, left,** and **right** properties.

Position



Paragraph 1 is absolute

Paragraph 2 is static

Paragraph 3 is relative

Notes:

- To use absolute position, you need to define **position** property on some ancestor
- **html** element by default defines position relative

Website Layout

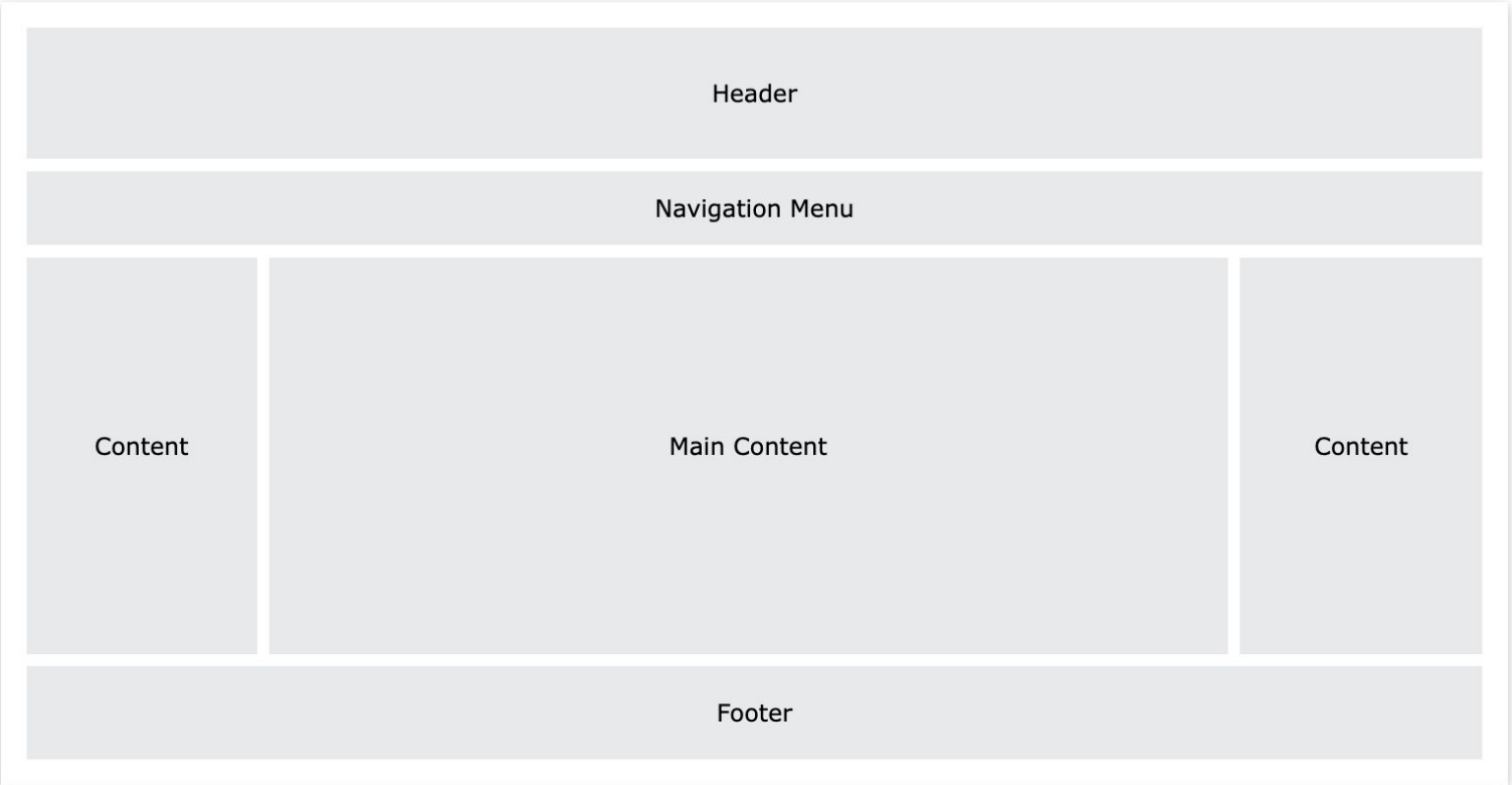
We can restructure our HTML document to have different layouts depending on:

- content (primary, secondary)
- semantics (header, navigation, content, footer)
- screen size (desktop, tablet, mobile)

Let’s have a look at a typical desktop layout

Website Layout

One of the typical layouts for big screen



Website Layout

Several ideas to mention:

- Use **semantic** elements (header, nav, footer, section)
- Header, nav and footer are **block** elements, i.e. take all line
- To make **multi-column** page:
  - use **flex** property on the containing element (section)
  - define column width in percentage (e.g.: 25%-50%-25%)
- Note: there are other solutions as well

Summary

- The Box Model:
  - Understand the structure of Box
  - Use overflow property to take care of extra content
  - Use box-sizing property carefully

- Positioning elements on the page can be done using:
  - Display
  - Float
  - Position

- Website layouts depend on several factors (content, screen)

bonus info