ECEN 4213
Embedded Computer System Design

HTML, CSS, JavaScript and Python

Instructor: Dr. Weihua Sheng, weihua.sheng@okstate.edu

TA: Zhanjie Chen, zhanjie.chen@okstate.edu

Claudia Pauyac, cpauyac@okstate.edu

Fall 2025

SCHOOL OF **ELECTRICAL** AND
COMPUTER ENGINEERING

I. HTML

II. CSS

III. JavaScript

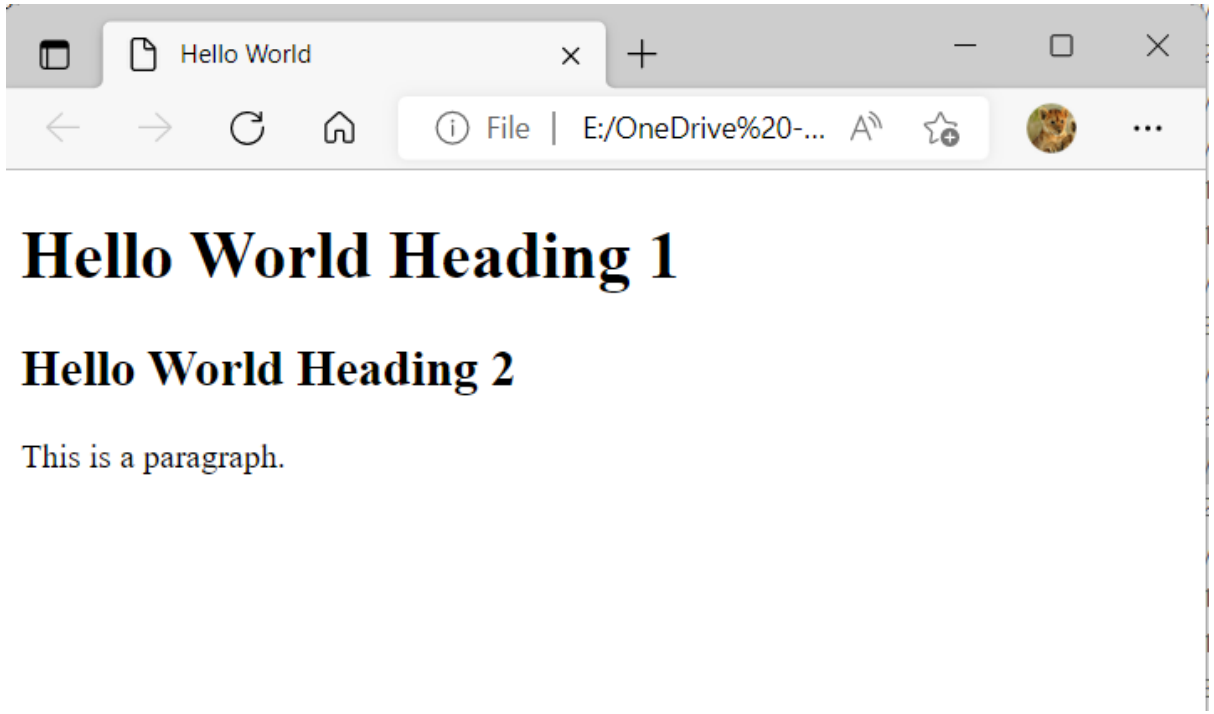
IV. Python

I. HTML - Overview

- **HTML: H**ypertext **M**arkup **L**anguage
- "Hypertext" refers to links that connect web pages to one another, either within a single website or between websites. Links are a fundamental aspect of the Web. By uploading content to the Internet and linking it to pages created by other people, you become an active participant in the World Wide Web.
- As its name suggests, HTML is a Markup Language which means you use HTML to simply "mark-up" a text document with tags that tell a Web browser how to structure it to display.

A Hello World Example

```
<!DOCTYPE html>
<html>

  <head>
      <title>Hello World</title>
  </head>

  <body>
    <h1>Hello World Heading 1</h1>
    <h2>Hello World Heading 2</h2>
    <p>This is a paragraph</p>
  </body>

</html>
```



Save the code in a file named *hello.html* and open it with a browser.

```html
<!DOCTYPE html>
<html>

  <head>
      <title>Hello World</title>
  </head>

  <body>
    <h1>Hello World Heading 1</h1>
    <h2>Hello World Heading 2</h2>
    <p>This is a paragraph</p>
  </body>

</html>
```

- The *<!DOCTYPE html>* declaration defines that this document is an HTML5 document
- The *<html>* element is the root element of an HTML page
- The *<head>* element contains meta information about the HTML page
- The *<title>* element specifies a title for the HTML page (which is shown in the browser's title bar or in the page's tab)
- The *<body>* element defines the document's body, and is a container for all the visible contents, such as headings, paragraphs, images, hyperlinks, tables, lists, etc.
- The *<h1>* and *<h2>* elements defines a large heading
- The *<p>* element defines a paragraph

Heading Tags

HTML provides six levels of headings: *<h1>*, *<h2>*, *<h3>*, *<h4>*, *<h5>*, and *<h6>*

```
<!DOCTYPE html>
<html>

  <head>
      <title>Hello World</title>
  </head>

  <body>
    <h1>Hello World Heading 1</h1>
    <h2>Hello World Heading 2</h2>
    <h3>Hello World Heading 3</h3>
    <h4>Hello World Heading 4</h4>
    <h5>Hello World Heading 5</h5>
    <h6>Hello World Heading 6</h6>
  </body>

</html>
```

Some concepts:

- An **HTML element** is defined by a starting tag. If the element contains other content, it ends with a closing tag, where the element name is preceded by a forward slash

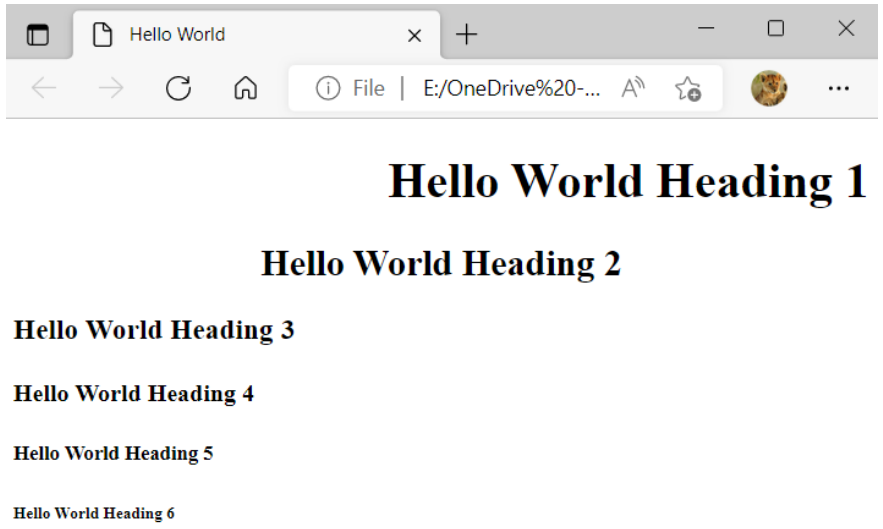
```
<h1>Hello World Heading 1</h1>

<head>
    <title>Hello World</title>
</head>
```

Start Tag	Content	End Tag
<h1>	Hello World Heading 1	</h1>
<head>	<title>Hello World</title>	</head>

There are some HTML elements which don't need to be closed, such as *<img.../>*, *<hr />* and *
* elements. These are known as void elements.

- An **attribute** is used to define the characteristics of an HTML element and is placed inside the element's opening tag. All attributes are made up of two parts: a name and a value:

```html
<!DOCTYPE html>
<html>
  <head>
      <title>Hello World</title>
  </head>
  <body>
      <h1 align="right">Hello World Heading 1</h1>
      <h2 align="center">Hello World Heading 2</h2>
      <h3 align="left">Hello World Heading 3</h3>
      <h4>Hello World Heading 4</h4>
      <h5>Hello World Heading 5</h5>
      <h6>Hello World Heading 6</h6>
  </body>
</html>
```



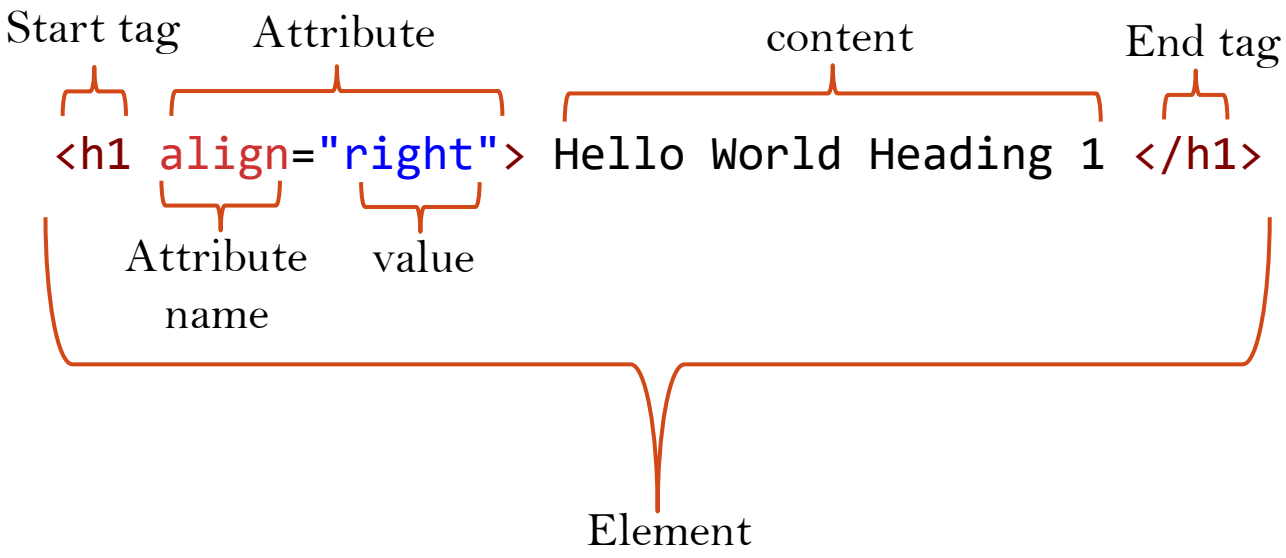
- The **name** is the property you want to set. For example, the heading *<h1>* element in the example carries an attribute whose name is align, which you can use to indicate the alignment of heading on the page.
- The **value** is what you want the value of the property to be set and always put within quotations. The below example shows three possible values of align attribute: **left**, **center** and **right**.

Generic attributes

Here's a table of some other attributes that are readily usable with many of the HTML tags.

Attribute	Options	Function
align	right, left, center	Horizontally aligns tags
valign	top, middle, bottom	Vertically aligns tags within an HTML element.
bgcolor	numeric, hexidecimal, RGB values	Places a background color behind an element
background	URL	Places a background image behind an element
id	User Defined	Names an element for use with Cascading Style Sheets.
class	User Defined	Classifies an element for use with Cascading Style Sheets.
width	Numeric Value	Specifies the width of tables, images, or table cells.
height	Numeric Value	Specifies the height of tables, images, or table cells.
title	User Defined	"Pop-up" title of the elements.

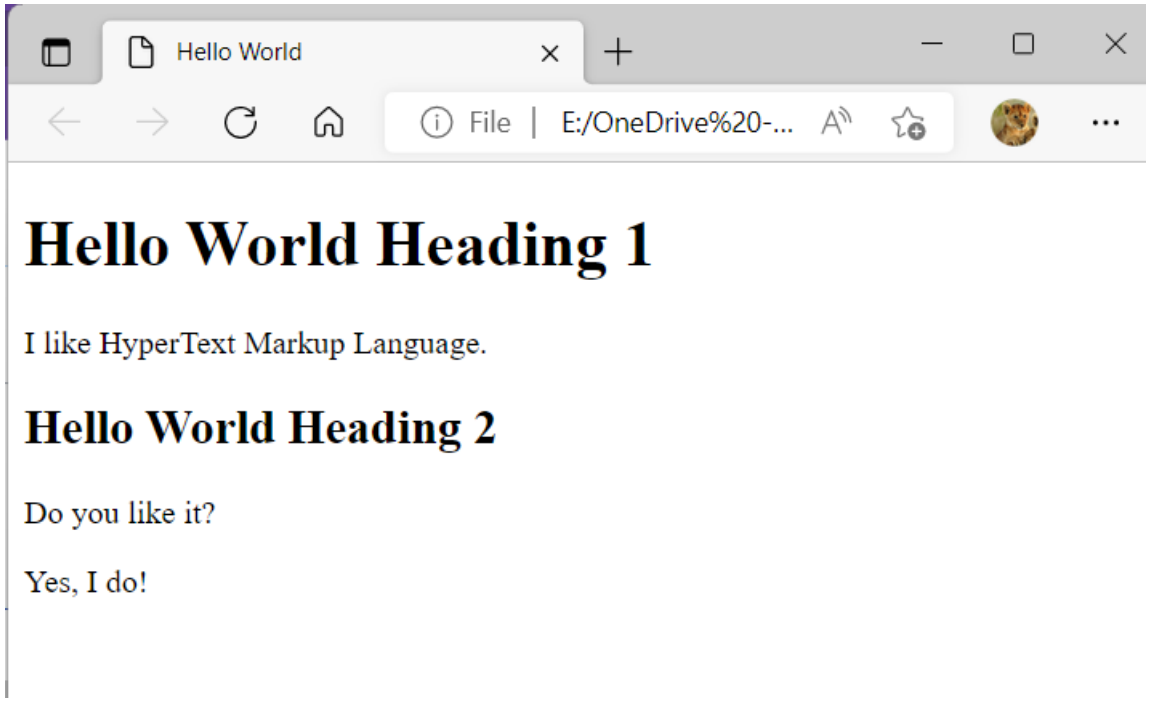
Some concepts:



Paragraph tag

The Paragraph Tag * * is used to structure the text into different paragraphs.

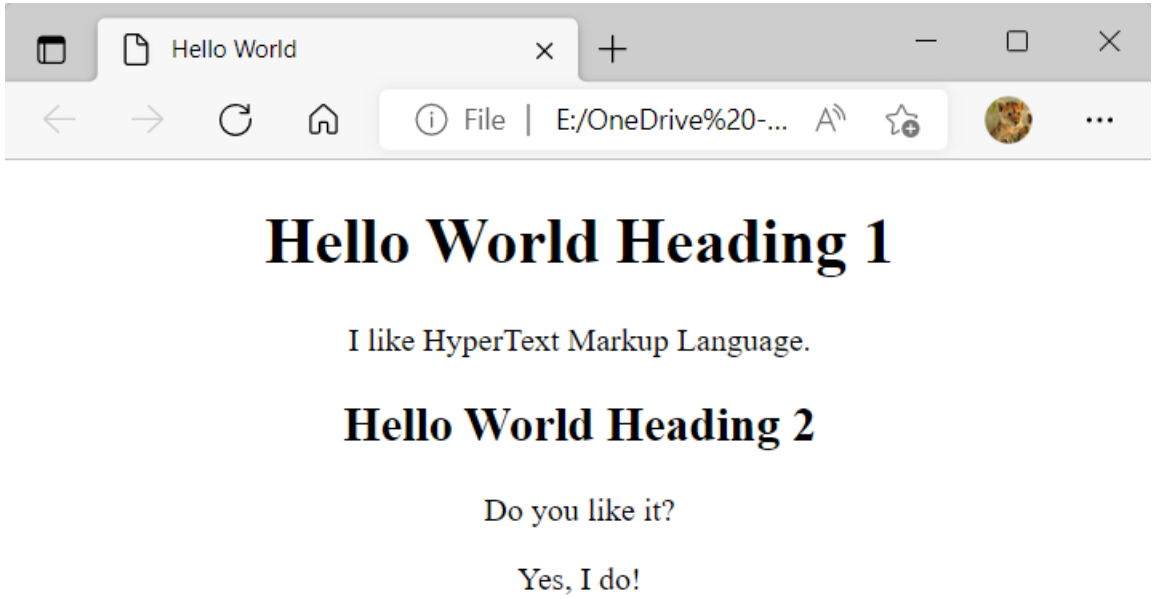
```
<!DOCTYPE html>
<html>

  <head>
      <title>Hello World</title>
  </head>

  <body>
    <h1>Hello World Heading 1</h1>
    <p>I like HyperText Markup Language.</p>
    <h2>Hello World Heading 2</h2>
    <p>Do you like it?</p>
    <p>Yes, I do!</p>
  </body>

</html>
```



Center tag

The center Tag * * put any content within the tag to center the content of the webpage.

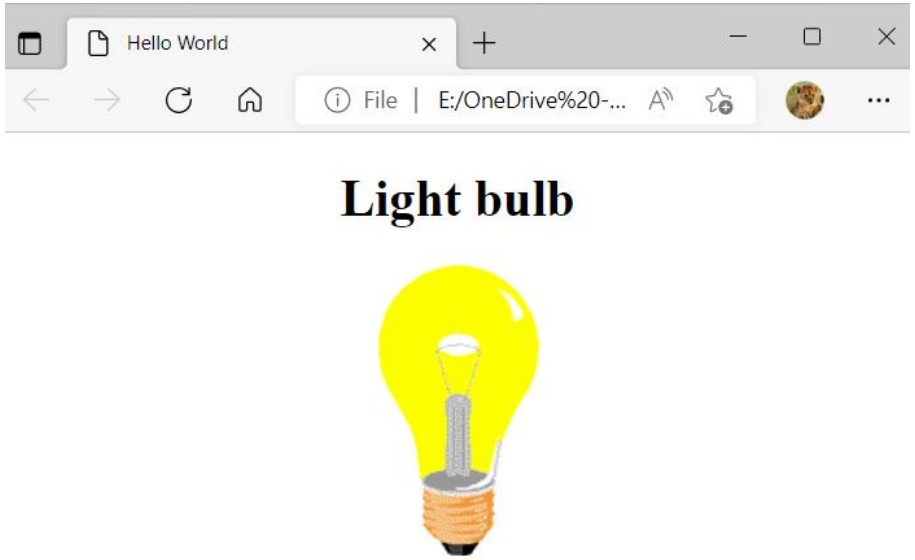
```
<!DOCTYPE html>
<html>

  <head>
      <title>Hello World</title>
  </head>

  <body>
    <center>
      <h1>Hello World Heading 1</h1>
      <p>I like HyperText Markup Language.</p>
      <h2>Hello World Heading 2</h2>
      <p>Do you like it?</p>
      <p>Yes, I do!</p>
    </center>
  </body>

</html>
```



Image

You can insert any image in your web page by using *<img>* tag.

```
<!DOCTYPE html>
<html>

  <head>
      <title>Hello World</title>
  </head>

  <body>
    <center>
      <h1 >Light bulb</h1>
      <img src="pic_bulbon.png" alt="bulb on">
    </center>
  </body>

</html>
```

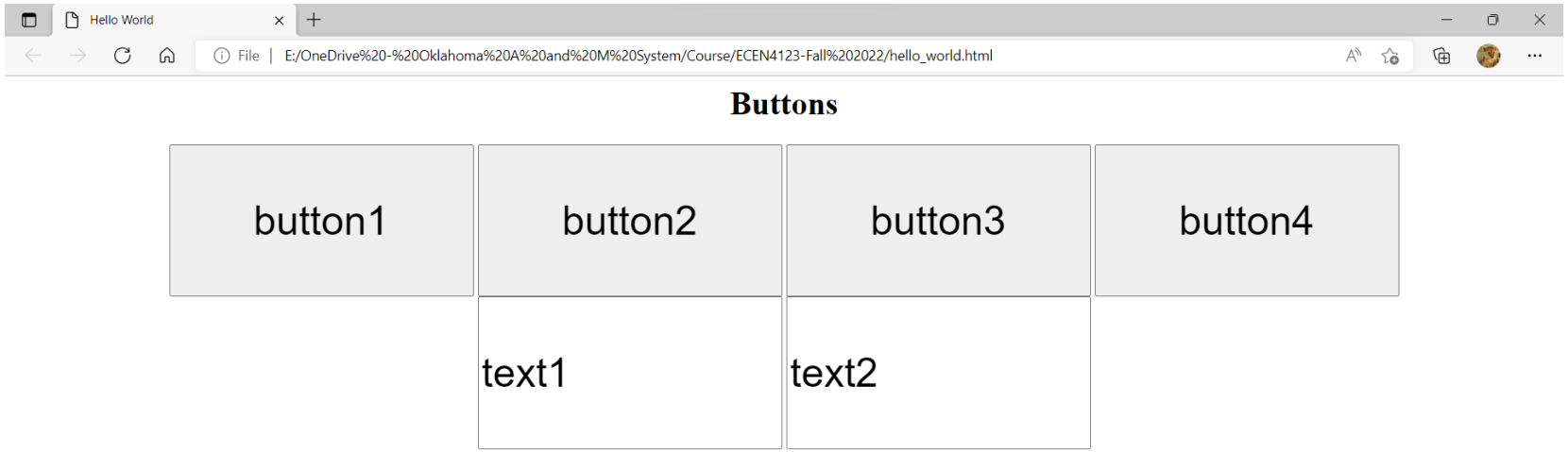


- You can use PNG, JPEG or GIF image file based on your comfort but make sure you specify correct image file name in **src** attribute. Image name is always case sensitive.
- The alt attribute specifies an alternate text for an image, if the image cannot be displayed.

Button

In HTML, the *<input>* can be specified using where a user can enter data. By specifying the type attribute, we can have different input types. If we define the "type" attribute as "button", we will create a button in the webpage.

```
<html>
  <head>
      <title>Hello World</title>
  </head>
  <body>
    <center>
      <h1 >Buttons</h1>
      <div>
        <input style="font-size:40;height:150; width:300;" type = "button" value = "button1" />
        <input style="font-size:40;height:150; width:300;" type = "button" value = "button2" />
        <input style="font-size:40;height:150; width:300;" type = "button" value = "button3" />
        <input style="font-size:40;height:150; width:300;" type = "button" value = "button4" />
      </div>
      <input style="font-size:40;height:150; width:300;" type = "text" value = "text1" />
      <input style="font-size:40;height:150; width:300;" type = "text" value = "text2" />
    </center>
  </body>
</html>
```



https://www.w3schools.com/html/default.asp
https://www.tutorialspoint.com/html/index.htm

I. HTML

II. CSS

III. JavaScript

IV. Python

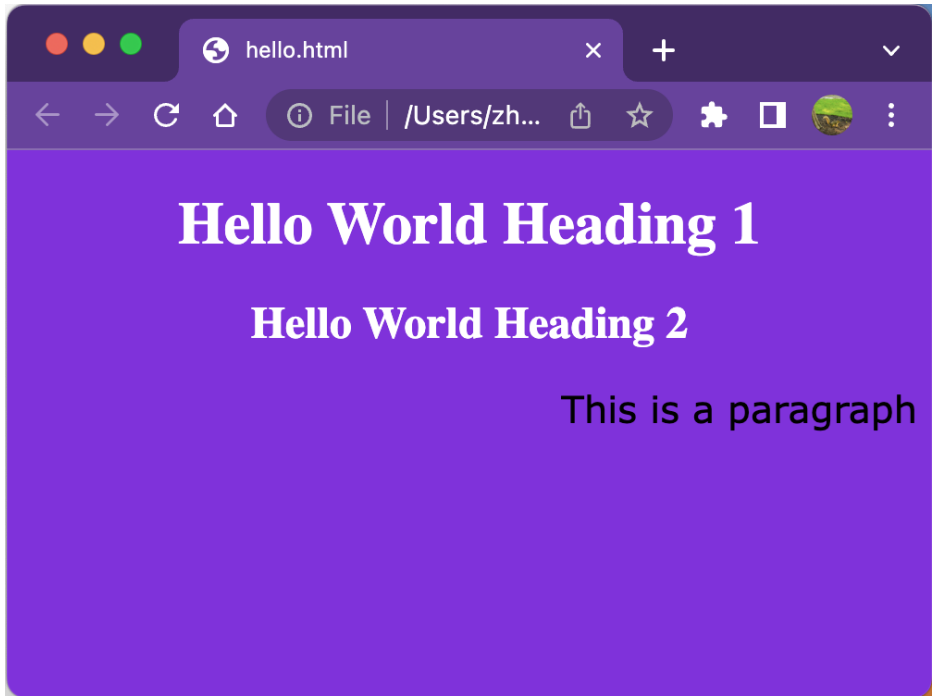
II. CSS - Overview

- CSS stands for Cascading Style Sheets.
- CSS is the language we use to style an HTML document.
- CSS describes how HTML elements should be displayed.

Internal CSS

There are three ways of inserting a CSS style sheet:

- Internal CSS
- External CSS
- Inline CSS




```
<!DOCTYPE html>
<html>
    <head>
        <style>
            body {
            background-color: blueviolet;
            }

            h1, h2 {
            color: white;
            text-align: center;
            }

            p {
            font-family: verdana;
            font-size: 20px;
            text-align: right;
            }
        </style>
    </head>

    <body>
        <h1>Hello World Heading 1</h1>
        <h2>Hello World Heading 2</h2>
        <p>This is a paragraph</p>
    </body>
</html>
```

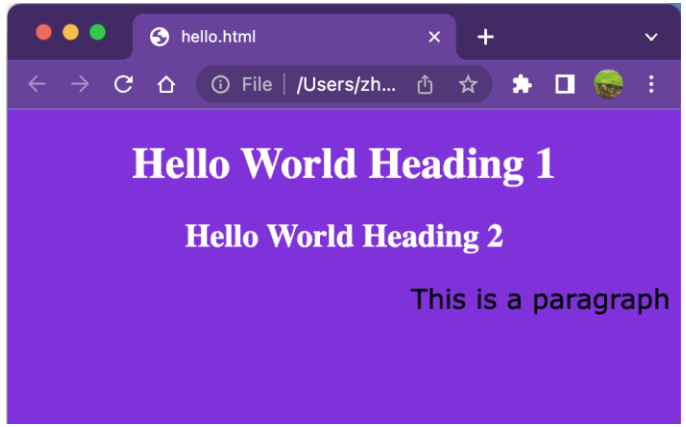
External CSS

hello.html

```html
<!DOCTYPE html>
<html>
    <head>
        <link rel="stylesheet" href="style.css">
    </head>
    <body>
        <h1>Hello World Heading 1</h1>
        <h2>Hello World Heading 2</h2>
        <p>This is a paragraph</p>
    </body>
</html>
```

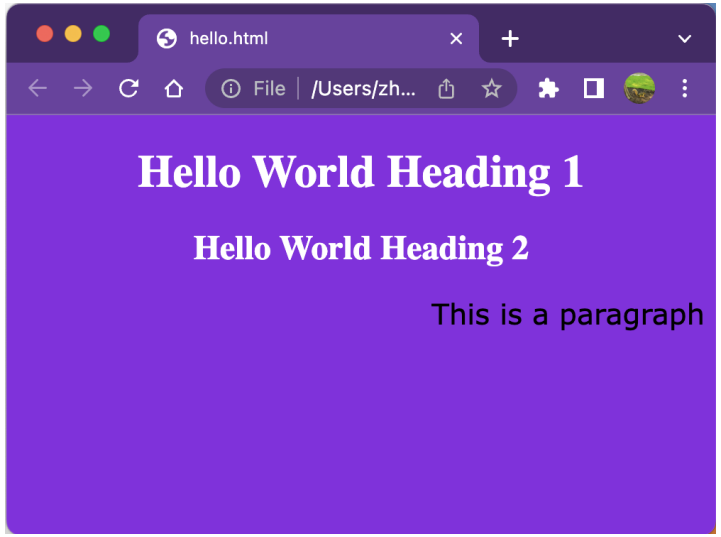
style.css

```css
body {
background-color: blueviolet;
}

h1, h2 {
color: white;
text-align: center;
}

p {
font-family: verdana;
font-size: 20px;
text-align: right;
}
```



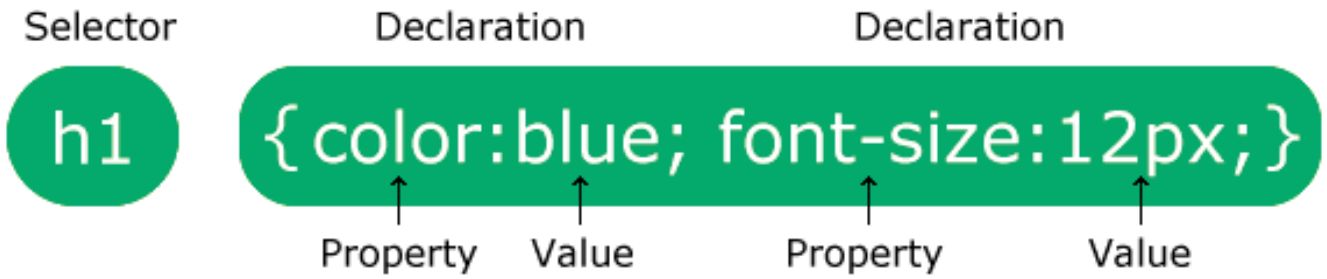
Inline CSS

```
<!DOCTYPE html>
<html>
        <head>
        </head>

        <body style="background-color:blueviolet;">
                <h1 style="color:white;text-align:center;">Hello World Heading 1</h1>
                <h2 style="color:white;text-align:center;">Hello World Heading 2</h2>
                <p style="font-family: verdana;font-size: 20px;text-align: right;">This is a paragraph</p>
        </body>
</html>
```



CSS Syntax

A CSS rule consists of a selector and a declaration block:



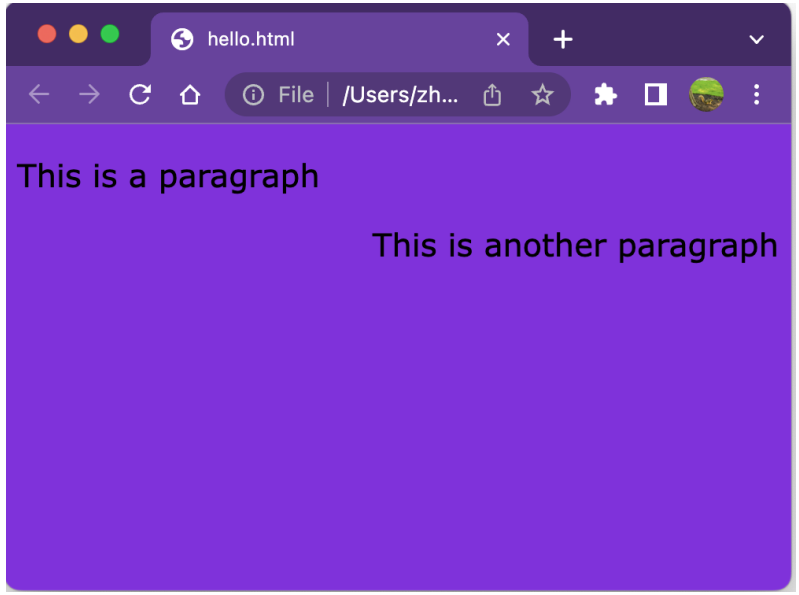
- The selector points to the HTML element you want to style.
- The declaration block contains one or more declarations separated by semicolons.
- Each declaration includes a CSS property name and a value, separated by a colon.
- Multiple CSS declarations are separated with semicolons, and declaration blocks are surrounded by curly braces.

https://www.w3schools.com/css/css_syntax.asp

Selector

- A CSS selector selects the HTML element(s) you want to style.
- We can use the tag, id or class name to select an element.

In this example, we define an id “para1” and style the corresponding element.




```
<!DOCTYPE html>
<html>
    <head>
        <style>
        body {
        background-color: blueviolet;
        }
        p {
        font-family: verdana;
        font-size: 20px;
        text-align: right;
        }
        #para1{
        font-family: verdana;
        font-size: 20px;
        text-align: left;
        }
        </style>
    </head>
    <body>
        <p id="para1">This is a paragraph</p>
        <p>This is another paragraph</p>
    </body>
</html>
```

Properties

Page 1 of 5 Quick Reference Guide FREE

Cascading Style Sheets (CSS 3)

BACKGROUND	
background	*background-image* *background-position* *background-size* *background-repeat* *background-attachment* *background-origin* *background-clip* *background-color*
background-attachment	scroll | fixed
background-break	bounding-box | each-box | continuous
background-clip	*length* % border-box | padding-box | content-box | no-clip
background-color	*color* transparent
background-image	*url* none
background-origin	border-box | padding-box | content-box
background-position	top left | top center | top right | center left | center center | center right | bottom left | bottom center | bottom right *x-% y-%* *x-pos y-pos*
background-repeat	repeat | repeat-x | repeat-y | no-repeat
background-size	*length* % auto | cover | contain

BORDER	
border	*border-width* *border-style* *border-color*
border-break	*border-width*

BORDER	
border-top	*border-top-width* *border-style* *border-color*
border-top-color	*border-color*
border-top-style	*border-style*
border-top-width	thin | medium | thick *length*
border-width	thin | medium | thick *length*
border-radius	*border-top-right-radius* *border-bottom-right-radius* *border-bottom-left-radius* *border-top-left-radius*
border-top-right-radius	*length*
border-bottom-right-radius	*length*
border-bottom-left-radius	*length*
border-top-left-radius	*length*
box-shadow	inset || [*length, length, length, length* || *<color>*] none
border-style	none | hidden | dotted | dashed | solid | double | groove | ridge | inset | outset

FONT	
font	*font-style* *font-variant* *font-weight* *font-size/line-height* *font-family* caption | icon | menu | message-box | small-caption | status-bar
font-family	*family-name* *generic-family* inherit

BOX MODEL	
float	left | right | none
height	auto *length* %
max-height	none *length* %
max-width	none *length* %
min-height	none | inherit *length* %
min-width	none | inherit *length* %
width	auto % *length*
margin	*margin-top* *margin-right* *margin-bottom* *margin-left*
margin-bottom	auto *length* %
margin-left	auto *length* %
margin-right	auto *length* %
margin-top	auto *length* %
padding	*padding-top* *padding-right* *padding-bottom*

https://cloud.netlifyusercontent.com/assets/344dbf88-fdf9-42bb-adb4-46f01eedd629/d7fb67af-5180-463d-b58a-bfd4a220d5d0/css3-cheat-sheet.pdf

I. HTML

II. CSS

III. JavaScript

IV. Python

III. JavaScript - Overview

- HTML to define the content of web pages
- CSS to specify the style of web pages
- JavaScript to program the behavior of web pages
  - In HTML, JavaScript code is inserted between *<script>* and *</script>* tags.

```
<script>
document.getElementById(”a_id").innerHTML = "JavaScript";
</script>
```

There are three ways of inserting a JS code:

- Internal JS
  - Scripts can be placed in the *<body>*, or in the *<head>* section of an HTML page, or in both.
- External JS
  - External scripts are practical when the same code is used in many different web pages.
  - JavaScript files have the file extension **.js**.
  - To use an external script, put the name of the script file in the **src** (source) attribute of a *<script>* tag

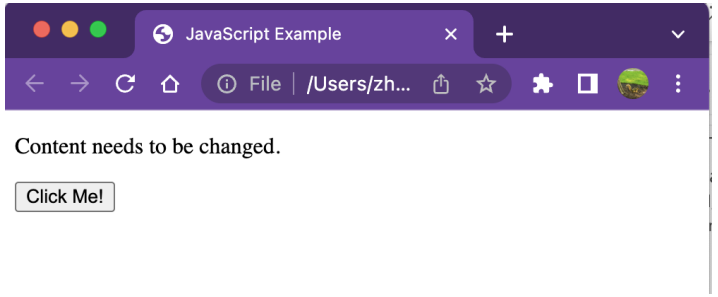
```
<script src=”JScode.js"></script>
<script src="http://code.jquery.com/jquery-latest.js"></script>
```

- Inline JS

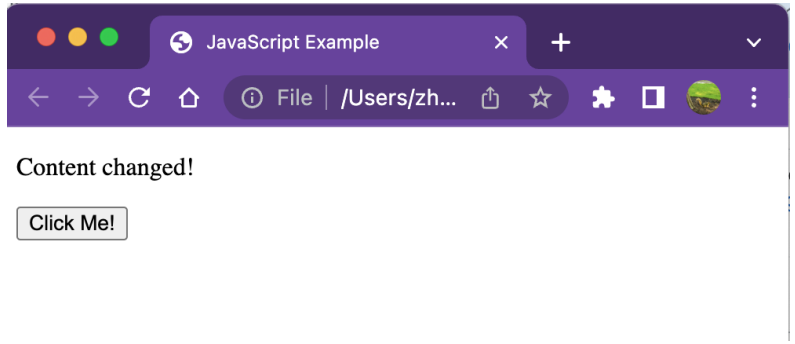
```
<input type="button" onclick="document.getElementById('p_id').innerHTML = 'Content changed!'" value="Click Me!"></input>
```

JavaScript Function

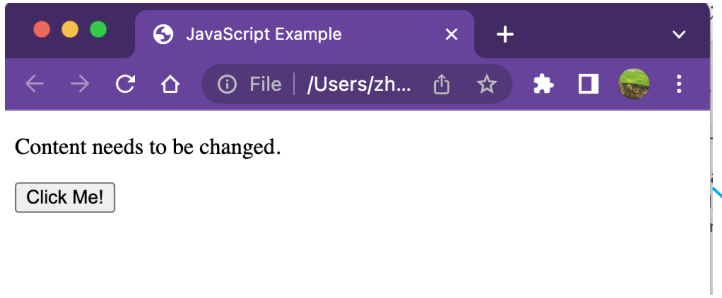
```
<!DOCTYPE html>
<html>
        <head>
                <title>JavaScript Example</title>
                <script>
                        function change_content(){
                                an_element = document.getElementById('p_id') // select an element
                                an_element.innerHTML = 'Content changed!'; // change the element content
                        }
                </script>
        </head>

        <body>
                <p id="p_id">Content needs to be changed.</p>
                <input type="button" onclick="change_content()" value="Click Me!"> // the “onclick” attribute is a JS event, when users click the button, it will call function
                “change_content()”.
        </body>
</html>
```



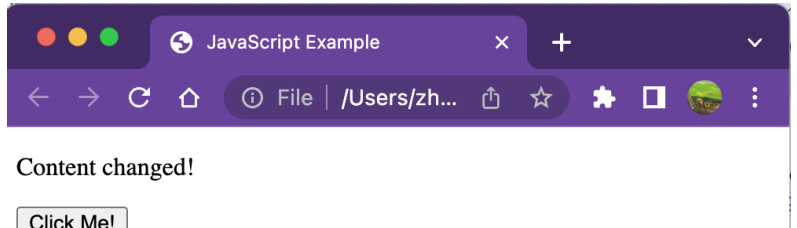
After clicking

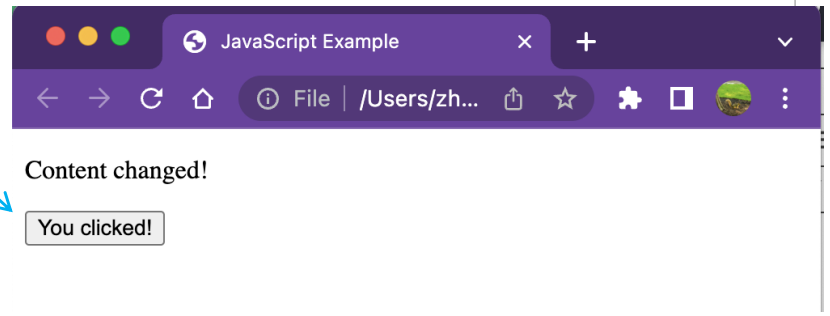


```
<script>
        function change_content(){
                an_element = document.getElementById('p_id'); // select an element
                an_element.innerHTML = 'Content changed!'; // change the element content
                button_element = document.getElementById('a_button'); // select the button element
                button_element.value = "You clicked!"; // change the element value
        }
</script>
<input type="button" id = "a_button" onclick="change_content()" value="Click Me!">
```



After clicking





https://www.w3schools.com/default.asp
https://www.tutorialspoint.com/html/index.htm
https://www.smashingmagazine.com/2009/07/css-3-cheat-sheet-pdf/

I. HTML

II. CSS

III. JavaScript

IV. Python

IV. Python - Overview

- High-Level, Interpreted, Object-Oriented Programming Language
- Runs on Windows, macOS, and most Linux distributions
- Features:
  - Clear and easy to understand syntax
  - Wide variety of applications
  - Good for beginners

beginner.py × beginner_function.py

beginner.py > ...

```python
# Brief: this code is used for python beginners.
# Run: python3 beginner.py

# a is an integer number
a = 10
# b is an integer number
b = 20

# c is an integer number, the sum of a and b. Here, we use the operator '+'
c = a + b

# output is a string,
output = 'Sum of a and b is:' + str(a) + '+' + str(b) + '=' + str(c)

# The print() function prints the specified message to the screen.
print(output)
```

How to program in Python

- Download Python from: http://python.org
- Python software can be developed in an Integrated Development Environment (IDE), which includes an editor, debugging tools, and other features
  - Visual Studio Code
- Python language grammar
  - Comment, Variables, Loops, If/Else statements, Function

beginner.py - Summe... pi pi@raspberrypi: ~ 22:25

beginner.py - SummerCamp - Visual Studio Code

File Edit Selection View Go Run Terminal Help

EXPLORER

OPEN EDITORS

SUMMERCAMP
- __MACOSX
- Final_Version
- Final_Version.1
- Lab2
- Robotics Summer Camp (Beginner Level)
- SummerCamp
- beginner_function.py
- beginner.py
- Lab1.py
- Robotics Summer Camp (Beginner Level)-20...
- SummerCamp.zip

OUTLINE

TIMELINE

beginner.py | beginner_function.py

beginner.py > ...

```python
# Brief: this code is used for python beginners.
# Run: python3 beginner.py

# a is an integer number
a = 10
# b is an integer number
b = 20

# c is an integer number, the sum of a and b. Here, we use the operator '+'
c = a + b

# output is a string,
output = 'Sum of a and b is:' + str(a) + '+' + str(b) + '=' + str(c)

# The print() function prints the specified message to the screen.
print(output)

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
pi@raspberrypi:~/Desktop/SummerCamp $
```

You have Docker installed on your system. Do you want to install the recommended extensions for it?

Install Show Recommendations

⊗ 0 ⚠ 0 Ln 18, Col 1 Spaces: 4 UTF-8 LF Python

Software

Filename: *beginner.py*

beginner.py × beginner_function.py

beginner.py > _

```python
# Brief: this code is used for python beginners.
# Run: python3 beginner.py

# a is an integer number
a = 10
# b is an integer number
b = 20

# c is an integer number, the sum of a and b. Here, we use the operator '+'
c = a + b

# output is a string,
output = 'Sum of a and b is:' + str(a) + '+' + str(b) + '=' + str(c)

# The print() function prints the specified message to the screen.
print(output)
```

Comment

Variables:

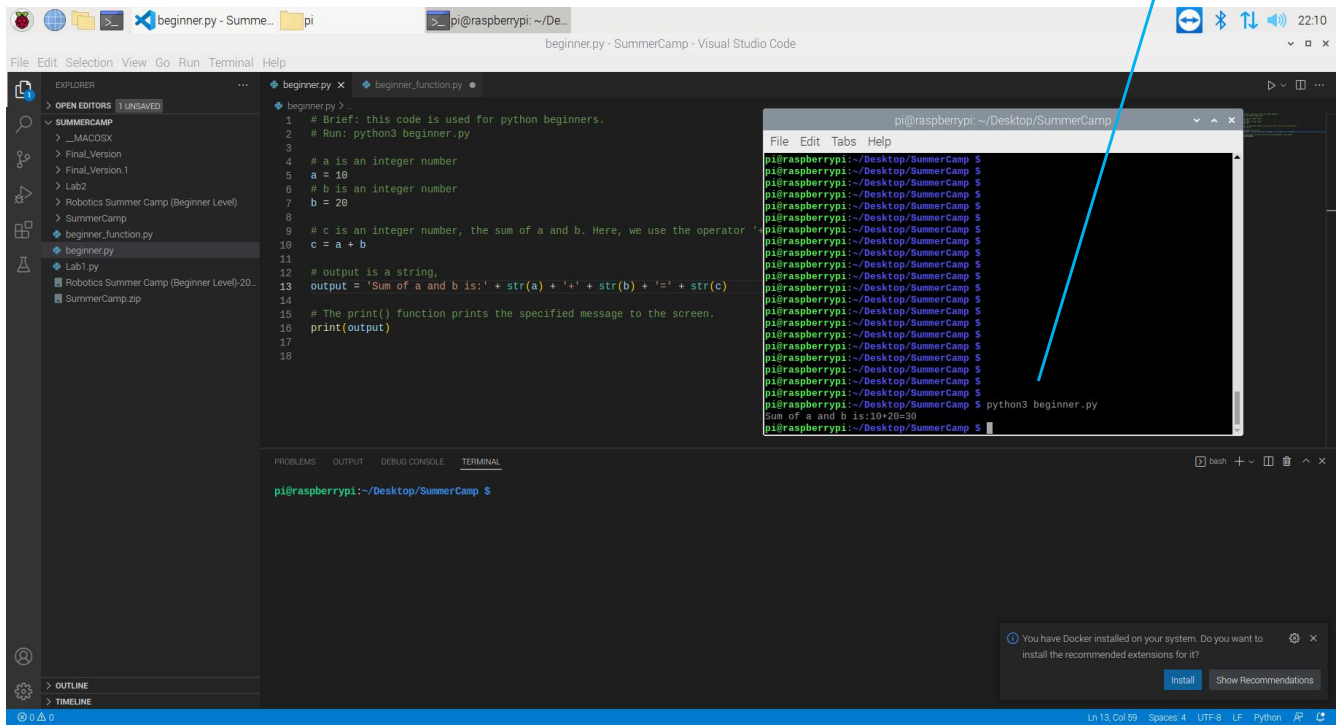
- Integer: a, b, c
- String: output

str() function,
Convert number to string

print() function,
Print messages on screen

Run: *python3 beginner.py*

```
pi@raspberrypi:~/Desktop/SummerCamp $ python3 beginner.py
Sum of a and b is:10+20=30
```



Declaring Variables

- Variables can be numbers, letters, or strings of text
- To declare a variable in Python:
  - *i = 5 #Creates a variable named 'i' with a value of 5*
  - *str_i = "Go Pokes!" #Creates a variable with the value "Go Pokes!"*
- The value associated with a variable can be changed:
  - *i = i+1 #Adds one to the previous value of i. What is the new value?*
  - *i = 9 #Sets the variable i to the value 9*

Functions

Function: Only write the code once, then call it as many times as you like

beginner.py beginner_function.py

beginner_function.py > ...

```python
# Brief: this code is used for python beginners.
# Run: python3 beginner_function.py

def sum_function(p_a, p_b):
    # c is an integer number, the sum of a and b. Here, we use the operator '+'
    c = p_a + p_b

    # output is a string,
    output = 'Sum of a and b is:' + str(p_a) + '+' + str(p_b) + '=' + str(c)

    # The print() function prints the specified message to the screen.
    print(output)

    # return the value
    return c


# call the function
d = sum_function(10, 20)
print('Get the result:', d)
```

def - define a function with any name you like (in this case *sum_function*)

(p_a, p_b) - inputs to function, to be used inside function

return - what the function outputs for you to use

Function must be defined before you use it in the rest of the code

Run: *python3 beginner_function.py*

```
pi@raspberrypi:~/Desktop/SummerCamp $ python3 beginner_function.py
Sum of a and b is:10+20=30
Get the result: 30
```

<u>Recall</u>: how to call a function

- If the functions are in the same file

  d = sum_function(10, 20)

- If the functions are from other modules, remember to import the modules first

  Example:

  - *import time*
  - *time.sleep(0.1)*

Loops

- Loops are used to perform a set of actions more than once, so that code can be reused rather than being written over and over.
- "While" loops: continue to loop until a condition is met
  - **Repeating a loop n number of times:** declare a variable that is equal to zero, and add one to the variable within the loop (example below). Executes code within the loop until the variable is equal to *n*, and then breaks out of the loop.
  - **Infinity loop:** *while(true)*, *while(1 == 1)*, or any statement that is always true

Condition

```
i = 0

while i < 5:
    print("Hello world!")
    i = i + 1
```

Output

```
Hello world!
Hello world!
Hello world!
Hello world!
Hello world!
```

Example of an infinite while loop

```
while True:
    print("Hello world!")
```

If/Else Statements

- If/Else statements provide a way to make a decision in your code based on a condition.
  - Checks if a value is greater than, less than or equal to another value
  - <u>Example</u>: Variable '*c*' is declared with a value of 5
    - *c == 5 # Evaluates to true*
    - *c > 7 # Evaluates to false*
    - *c < 10 # Evaluates to true*

```
a = 2020;
b = 2022;

if b == 2022:
    print("The year is 2022")
else:
    print("The year is not 2022")

if a > b:
    print("2020 is greater than 2022")
if b > a:
    print("2022 is greater than 2020")
```

Output:

```
The year is 2022
2022 is greater than 2020
```

Resources

- https://python.org
- https://raspberrypi.org
- https://linuxhint.com/raspberry-pi-history/
- https://www.w3schools.com/python
- https://www.ics.com/blog/control-raspberry-pi-gpio-pins-python