WiFi & AP Mode

ECE 476 Advanced Embedded Systems

Jake Glower - Lecture #32

Please visit Bison Academy for corresponding
lecture notes, homework sets, and solutions

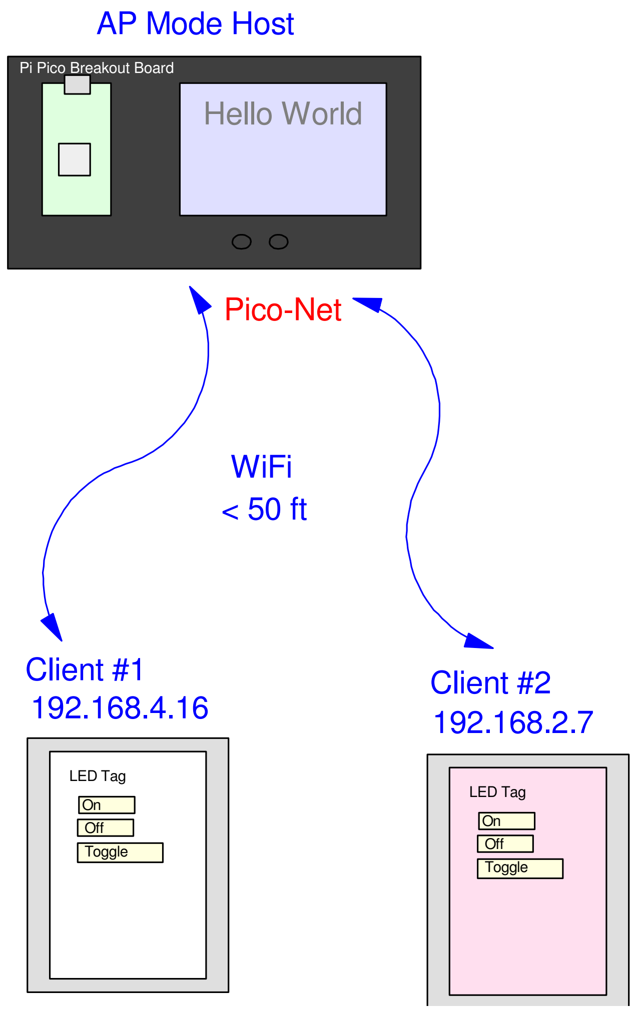
Introduction:

The Pi-Pico W has WiFi capabilities.

- You can create your own network
  - AP Mode
  - Range about 50 feet
  - This lecture
- You can access a wireless network
  - Range about 300 feet outdoors
  - Future lecture

This lecture looks at

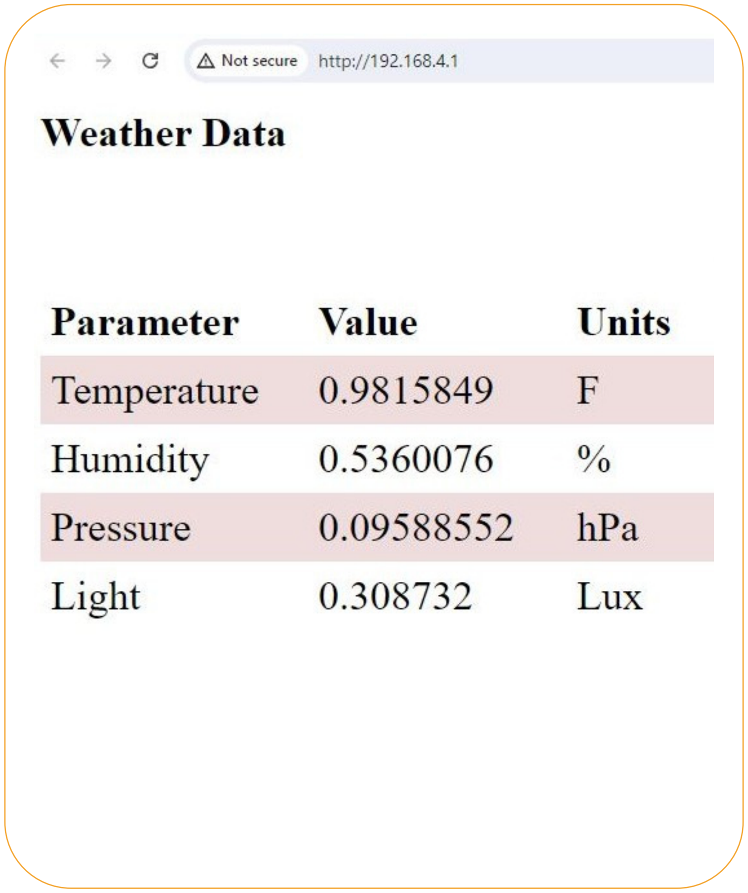
- Creating a stand-alone wireless local area network (wlan)
- Creating web pages
- Displaying information on these web pages



Where to go for help

Much of this information in this lecture comes from

- https://www.youtube.com/watch?v=cZNoXXIEPbg
- https://medium.com/@shilleh/creating-a-wireless-network-with-raspberry-pi-pico-w-part-1-c896211f2bd6
- https://www.w3schools/html/



Weather Data

Parameter	Value	Units
Temperature	0.9815849	F
Humidity	0.5360076	%
Pressure	0.09588552	hPa
Light	0.308732	Lux

Creating a Local Network

Let's start with creating a local network

- Contains a single page that says *Hello World*
- Page is defined by routine *web_page()*
- Coding is html

One way to create a web page is with a text string

- note: html coding ignores double spaces and carriage returns
- Web page is a long run-on string
- (more on coding later)

```
import network
import time
import socket

def web_page():
  x = "<html><body><h1>Hello World</h1></body></html>"
  return(x)
```

Creating a Web Page (take 2)

Anther option

- Create a separate file on Pico board
- Add indentation, carriage returns as desired
- (easier to read)

```
<html>
<body>
<h1>Hello World</h1>
</body>
</html>
```

Python Code: *web_page()*

- Read in this file
- Remove the carriage returns
- Return the file as a string

```
def web_page():
    f = open("HelloWorld.html","rt")
    x = f.read()
    x = x.replace('\r\n',' ')
    return(x)
```

Creating a Wireless Local Area Network (WLAN)

Step 1: Define the network's name and password.

- *network.WLAN* creates a local area network
- *config()* sets the network name and password
- *active(True)* starts the process of activating the LAN

```
ssid = 'Pico-Network'
password = 'PASSWORD'

ap = network.WLAN(network.AP_IF)
ap.config(essid=ssid, password=password)
ap.active(True)

while ap.active() == False:
    pass
print('AP Mode Is Active, You can Now Connect')
print('IP Address To Connect to:: ' + ap.ifconfig()[0])
```

Creating a WLAN (step 2):

Once the LAN is active,

- *socket.socket()* creates a new socket for this network
- *bind()* locks in the address for this web page
- *listen(5)* determines how many devices can connect to this LAN
  - five in this case

```
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.bind(('', 80))
s.listen(5)
```

Once active, lock the address and allow five clients

At this point, you can now receive and respond to pings from devices

Step 3: Wait for a ping

s.accept() waits until you get a query

- such as hitting refresh

This returns two parameters

- *conn* The status of the connection
- *addr* The address of the device who sent the message.
  - Note the second byte is a counter.

```
conn, addr = s.accept()
print('conn = ', conn)
print('addr = ', addr)
print('Got a connection from %s' % str(addr))
```

shell

```
conn = <socket state = 3 timeout=-1 incoming=2000d1d8 off=0>
addr = ('192.168.4.16', 57986)
Got a connection from 192.168.4.16
```

Step 4: Send html code

Once you get a ping

- Send a web page back to the client
  - html code
- Close the connection

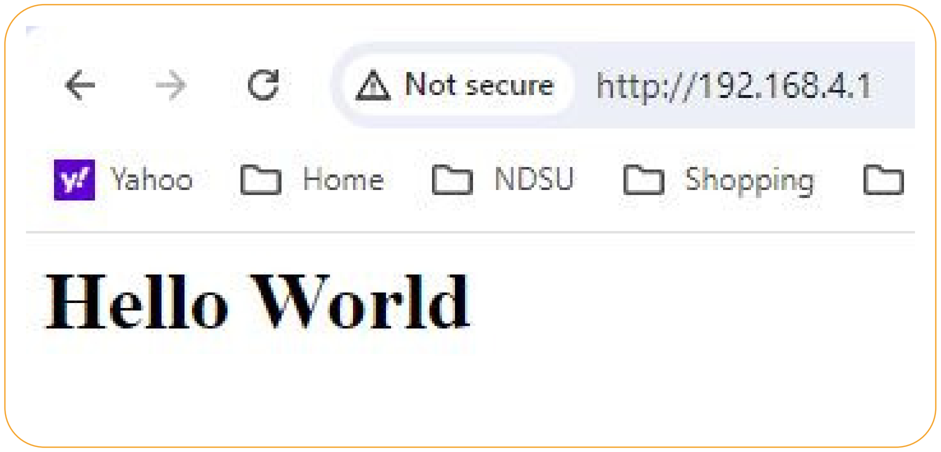
```
response = web_page()
conn.send(response)
conn.close()
```

The whole program looks like the following:

```
import network, time, socket

def web_page():
    f = open("HelloWorld.html","rt")
    x = f.read()
    x = x.replace('\r\n',' ')
    return(x)

ssid = 'Pico-Network'
password = 'PASSWORD'

ap = network.WLAN(network.AP_IF)
ap.config(essid=ssid, password=password)
ap.active(True)

while ap.active() == False:
    pass
print('AP Mode Is Active, You can Now Connect')
print('IP Address To Connect to:: ' + ap.ifconfig()[0])

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.bind(('', 80))
s.listen(5)

while(1):
    conn, addr = s.accept()
    print('Got a connection from %s' % str(addr))
    request = conn.recv(1024)
    print('Content = %s' % str(request))
    response = web_page()
    conn.send(response)
    conn.close()
```

If you look for WiFi network, you should see Pico-Network

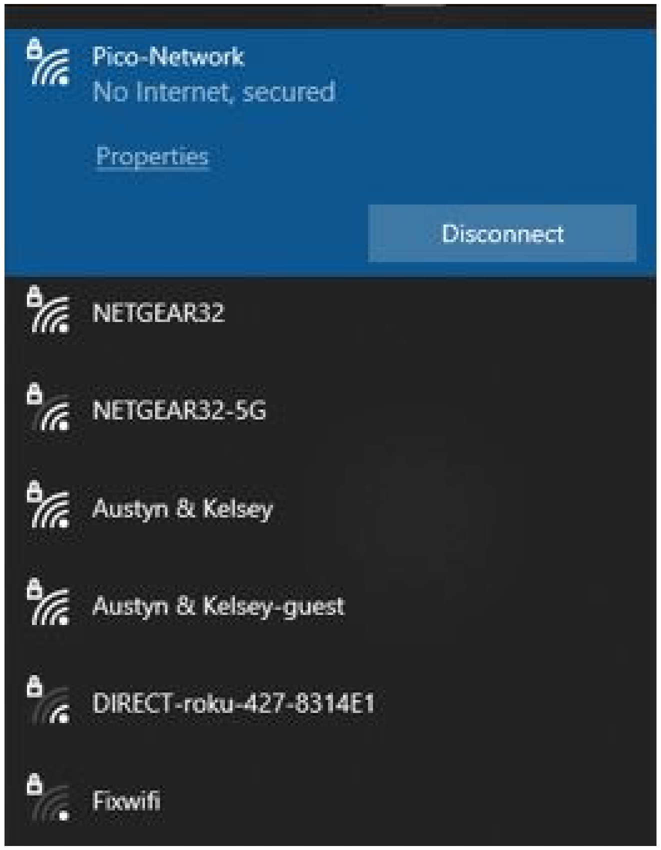
If you connect to web page 192.168.4.1, you will see the html image



Not secure http://192.168.4.1

Yahoo Home NDSU Shopping

Hello World



Pico-Network
No Internet, secured
Properties
Disconnect
NETGEAR32
NETGEAR32-5G
Austyn & Kelsey
Austyn & Kelsey-guest
DIRECT-roku-427-8314E1
Fixwifi

Shell Window

You will also see the reply from the connection in the shell window.

- Doesn't mean much here
- Will be used later on to pass data

```
AP Mode Is Active, You can Now Connect
IP Address To Connect to:: 192.168.4.1
Got a connection from ('192.168.4.16', 41178)

Content = b'GET / HTTP/1.1\r\nHost: 192.168.4.1\r\nConnection:
keep-alive\r\nCache-Control:
max-age=0\r\nUpgrade-Insecure-Requests: 1\r\nUser-Agent:
Mozilla/5.0 (Linux; Android 10; K) AppleWebKit/537.36 (KHTML,
like Gecko) Chrome/127.0.0.0 Mobile Safari/537.36\r\nAccept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/av
if,image/webp,image/apng,*/*;q=0.8,application/signed-exchange
;v=b3;q=0.7\r\nAccept-Encoding: gzip,
deflate\r\nAccept-Language: en-US,en;q=0.9\r\n\r\n'
```

Shell window if everything goes well

HTML Coding

Backing up a bit, in the previous example, html code was used to display *Hello World:*

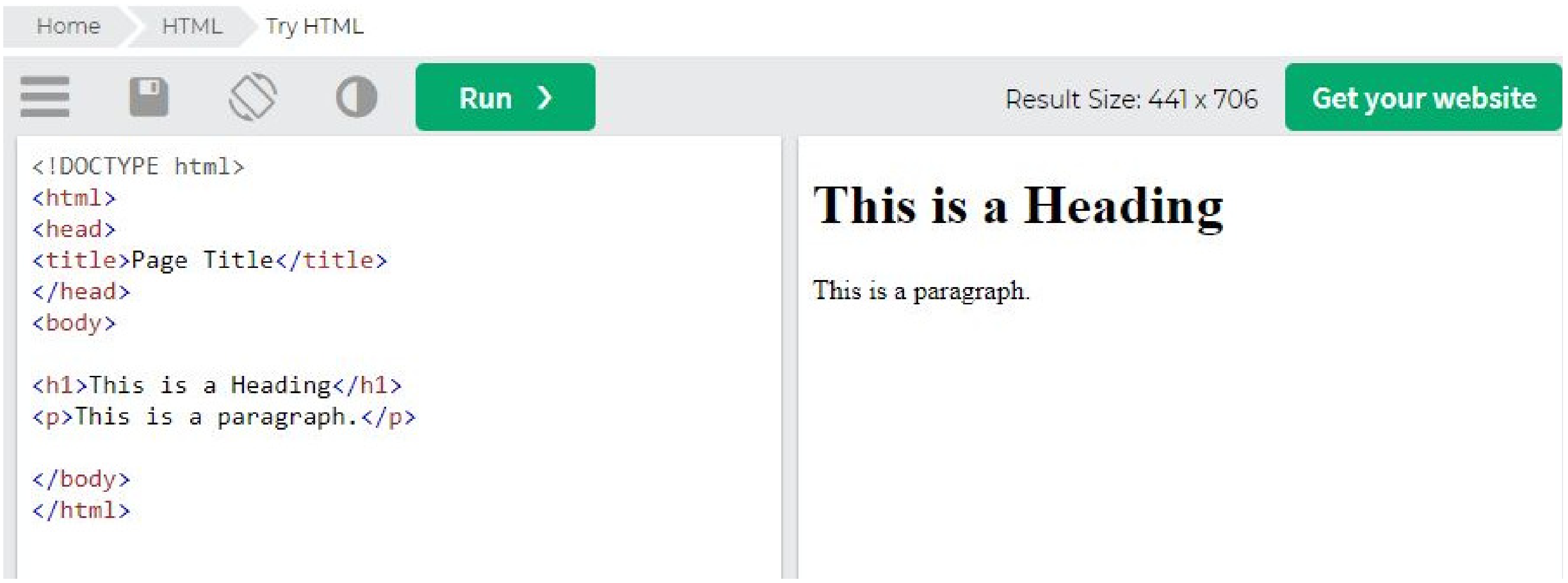
```
<html>
<body>
<h1>Hello World</h1>
</body>
</html>
```

You can do a lot more than this with html coding. You can even take several courses on html programming.

www.w3schools.com/html/

A good place to go for learning html coding is w3schools.

- Contains several lessons on html programming
- Also contains interactive windows
  - You can test out your code:



```
<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
</head>
<body>

<h1>This is a Heading</h1>
<p>This is a paragraph.</p>

</body>
</html>
```

This is a Heading

This is a paragraph.

www.w3schools.com/html/

Note on html syntax:

- html is not case sensitive
- html ignores carriage returns
- Single quotes and double quotes are interchangeable

For example, to create a string which contains quote symbols, you could use

```
x = "To quote Charlie Brown, 'Rats.'"
```

is the same as

```
x = 'To quote Charlie Brown, "Rats."'
```

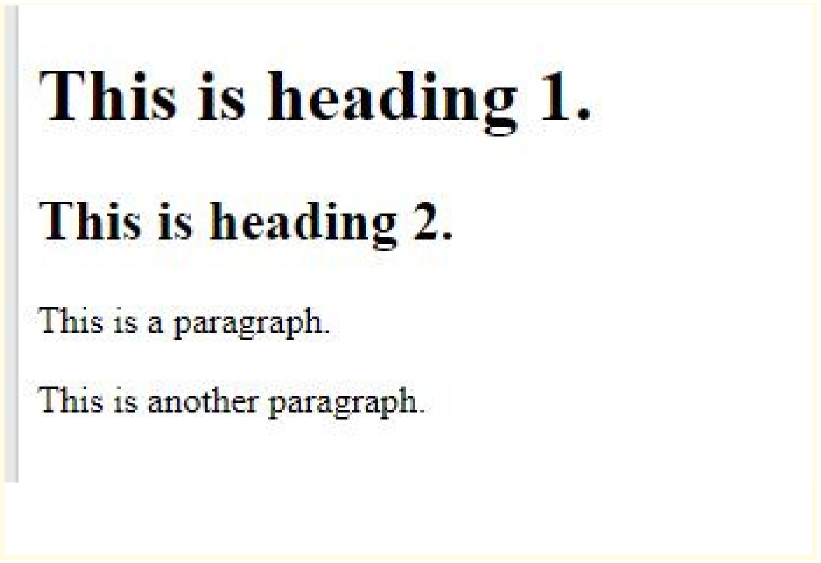
Here, we'll just go over creating a web page with

- headings,
- paragraphs, and
- a table.

The basic format for a html page is as follows:

- Sections start with a **<>**
- End of section is denoted with a back-slash

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

<h1>This is heading 1.</h1>
<h2>This is heading 2.</h2>

<p>This is a paragraph.</p>
<p>This is another paragraph.</p>

</body>
</html>
```



This is heading 1.

This is heading 2.

This is a paragraph.

This is another paragraph.

html options:

Some of the things you can add to his file are as follows:

Adding a link

```
<a href="http2://www.w3schools.com">This is a link</a>
```

Adding a carriage return

```
<br>
```

Hyperlink <a>

more on this later

Style: Set the color

```
<p style="color:red;">
```

Style: Set the font size

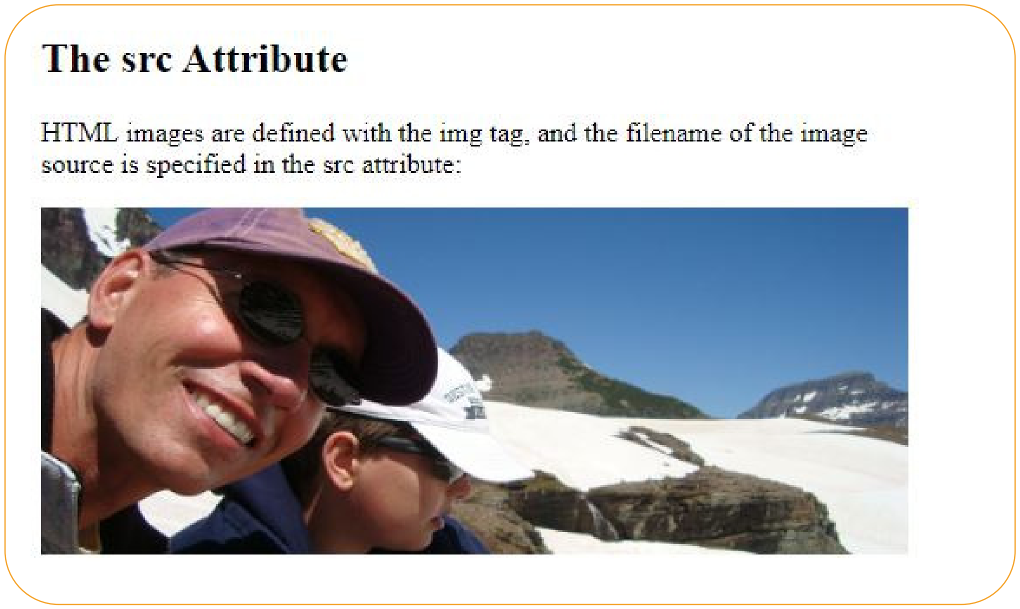
```
<p style="font-size:20px;">Paragraph in 20 point font.<\p>
<p style="font-size:300%;">Paragraph 300% font.<\p>
```

Style: Set background color

```
<body style="background-color:powderblue;">
<h1 style="background-color:tomato;">Heading</h1>
```

Adding an image <src>

```
<img
src="https://www.bisonacademy.com/uploads/3/4/4/0/34406028/glacier2_o
rig.hjpg" width="500" height="200">
```



The src Attribute

HTML images are defined with the img tag, and the filename of the image source is specified in the src attribute:

adding an image to a web page

Alternate text <alt>

If the image can't be displayed, the text to display instead

```
alt="Glacier NP"
```

Style: Font

```
<h1 style="font-family:ariel;">This is a heading</h1>
```

- Some fonts available include

- Arial	'Twas brillig and the slighy toves
- **Arial Black**	**Did gyre in the gimple in the wabe**
- Comic Sans	All mimsy were the borogoves
- Courier	And the mome rathe outgrabe.,
- Georgia	"Beware the jabberwock, my son!
- Helvetica	The jaws that bite, the claws that catch!
- **Imact**	**Beware the Jubjub bird, and shun,**
- Palatino	The frumious Bandersnatch!"
- Tahoma	He took his vorpal sword in hand;
- Trebuchet MS	Long time the manxome foe he sought--
- Times New Roman	So rested he by the Tumtum tree
- Verdana	And stood a while in thought

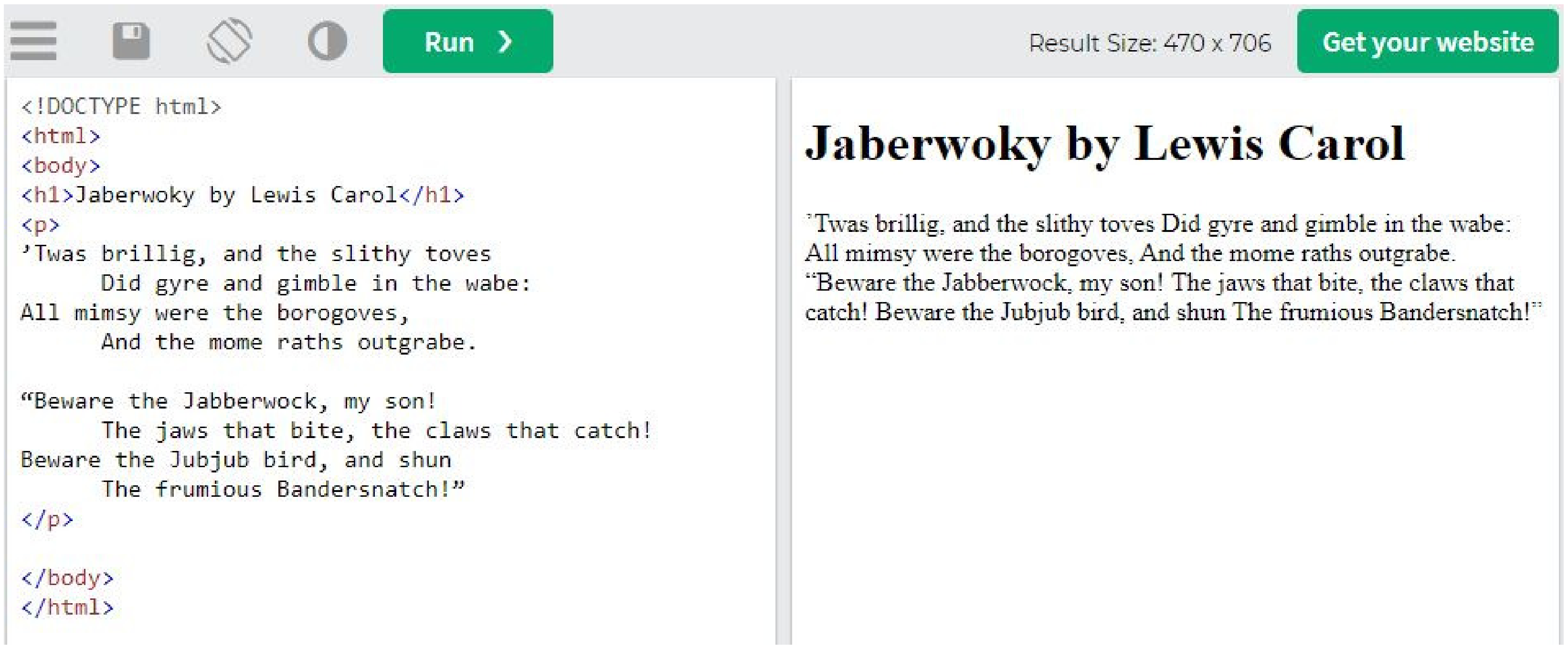
Jaberworky by Lewis Carol

Style: Text Align

```
<p style="text-align:center;">Centered paragraph.<\p>
options: left, center, right
```

Paragraphs <p>

- double spaces, carriage returns are ignored
  - have no effect on the resulting display



```
<!DOCTYPE html>
<html>
<body>
<h1>Jaberwoky by Lewis Carol</h1>
<p>
’Twas brillig, and the slithy toves
      Did gyre and gimble in the wabe:
All mimsy were the borogoves,
      And the mome raths outgrabe.

“Beware the Jabberwock, my son!
      The jaws that bite, the claws that catch!
Beware the Jubjub bird, and shun
      The frumious Bandersnatch!”
</p>

</body>
</html>
```

Jaberwoky by Lewis Carol

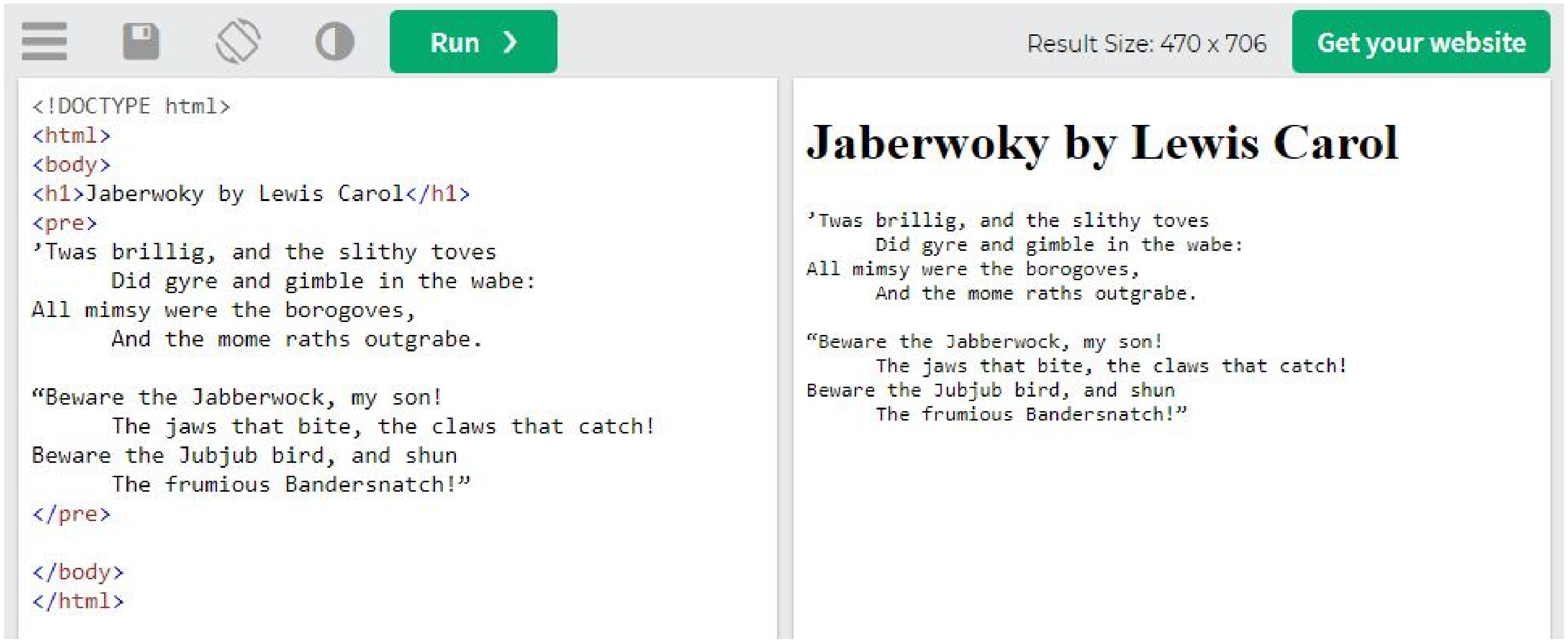
’Twas brillig, and the slithy toves Did gyre and gimble in the wabe: All mimsy were the borogoves, And the mome raths outgrabe. “Beware the Jabberwock, my son! The jaws that bite, the claws that catch! Beware the Jubjub bird, and shun The frumious Bandersnatch!”

Horizontal Rule <hr>

- draw a horizontal line

Preformatted Text <pre> <\pre>

- <p> ignores carriage returns and spaces.
- <pre> preserves carriage returns and spaces.



```
<!DOCTYPE html>
<html>
<body>
<h1>Jaberwoky by Lewis Carol</h1>
<pre>
’Twas brillig, and the slithy toves
      Did gyre and gimble in the wabe:
All mimsy were the borogoves,
      And the mome raths outgrabe.

“Beware the Jabberwock, my son!
      The jaws that bite, the claws that catch!
Beware the Jubjub bird, and shun
      The frumious Bandersnatch!”
</pre>

</body>
</html>
```

Jaberwoky by Lewis Carol

```
’Twas brillig, and the slithy toves
      Did gyre and gimble in the wabe:
All mimsy were the borogoves,
      And the mome raths outgrabe.

“Beware the Jabberwock, my son!
      The jaws that bite, the claws that catch!
Beware the Jubjub bird, and shun
      The frumious Bandersnatch!”
```

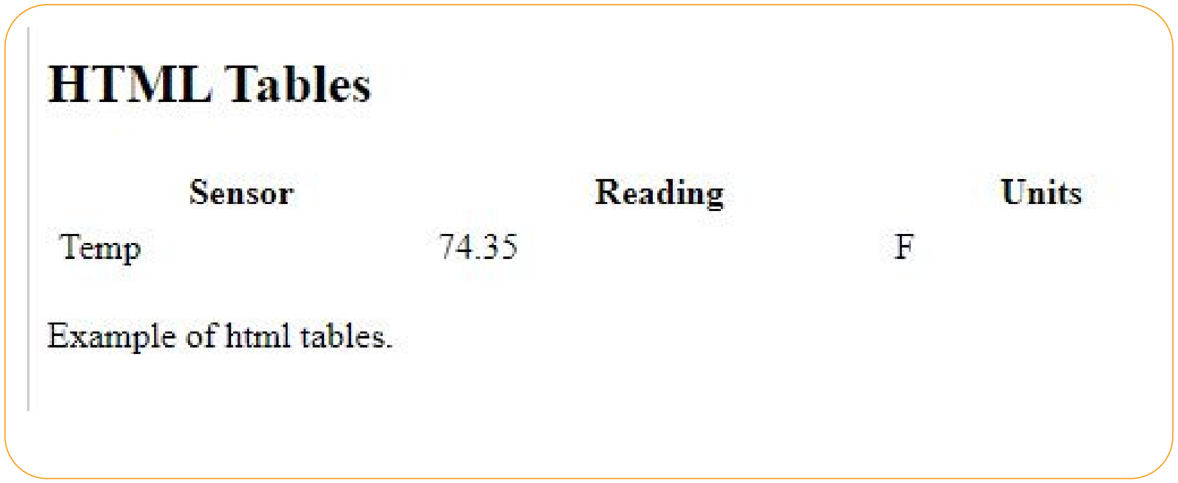
Formatting Text. Each of these are terminated with a back-slash (<b> ---- </b>)

• <b>	bold face
• <strong>	also bold face
• <i>	italic
• <mark>	marked text
• <small>	smaller text
• <del>	deleted text
• <sub>	subscript
• <sup>	superscript

Tables

Tables are a nice way to present information

```
<table>                         start of table
  <tr>                          start of row
    <th>Sensor</th>             table heading, column #1
    <th>Reading</th>
    <th>Units<th>
  </tr>                         end of first row
  <tr>                          start of second row
    <td>Temp</td>               table data
    <td>74.35</td>
    <td>F</td>
  ></tr>                        end of second row
</table>                        end of table

<p>Example of html table</p>
```



HTML Tables

Sensor	Reading	Units
Temp	74.35	F

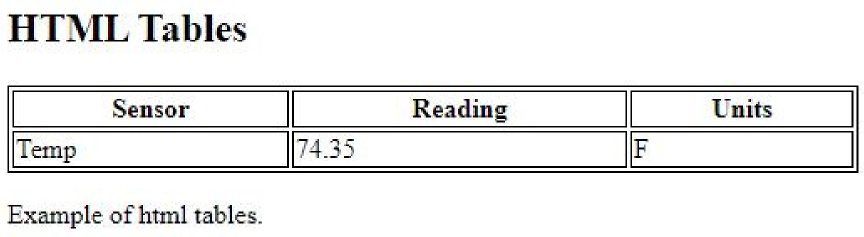
Example of html tables.

Borders

To add borders to a table, use the *border* statement.

- This adds a 1 pixel solid black border to
  - the table,
  - all rows, and
  - all data cells

```
table, th, td {
  border: 1px solid black
}
```



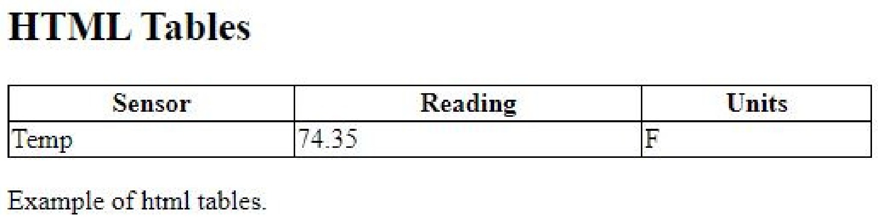
HTML Tables

Sensor	Reading	Units
Temp	74.35	F

Example of html tables.

collapse combines table / row / cell borders

```
table, th, td {
  border: 1px solid black;
  border-collapse: collapse;
}
```



HTML Tables

Sensor	Reading	Units
Temp	74.35	F

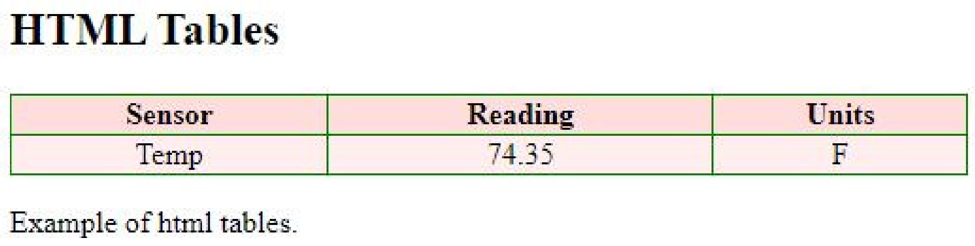
Example of html tables.

Row Colors

Color can be added to the table.

- Color is a 24-bit number
  - red - green - blue:

```
table, th, td {
  border: 1px solid white;
  border-collapse: collapse;
}
th, td {
  border-color: #008800;
  }
th {
  background-color: #FFDDDD;
  }
td {
  background-color: #FFEEEE;
  }
```



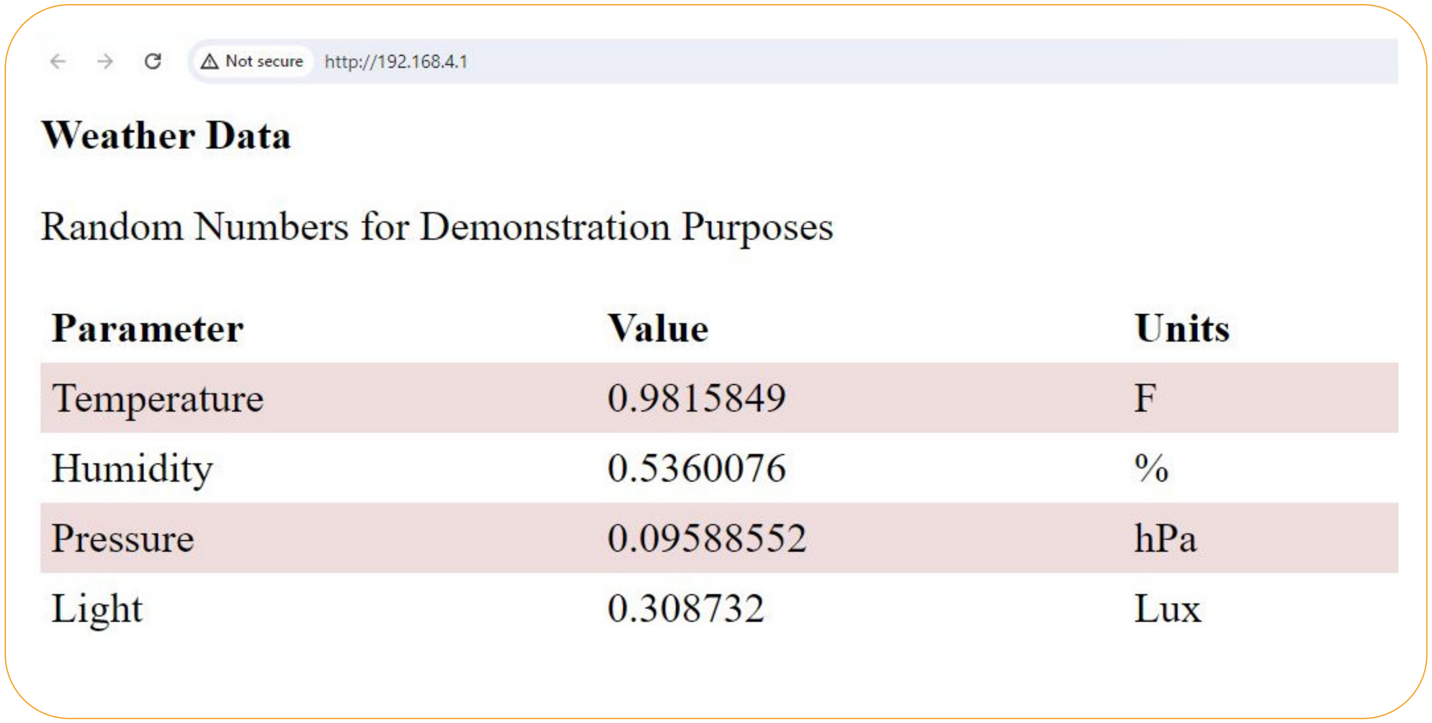
HTML Tables

Sensor	Reading	Units
Temp	74.35	F

Example of html tables.

Displaying Data in a Table

Suppose you want to generate a display where the values change based upon current readings:



Not secure http://192.168.4.1

Weather Data

Random Numbers for Demonstration Purposes

Parameter	Value	Units
Temperature	0.9815849	F
Humidity	0.5360076	%
Pressure	0.09588552	hPa
Light	0.308732	Lux

Example of displaying live data in a table

Displaying Data: One Option:

- Use dummy variables for the data
  - aaaaa, bbbbb, ccccc, ddddd in this example

```
<!DOCTYPE html><html>
<head>
  <style>
    table {  border-collapse: collapse;   width: 80%; }
    th, td {  text-align: left;  padding: 8px; }
    tr:nth-child(even) { background-color: #EEDDDD; }
    th, td, p, h2 { font-size:200%; }
    </style>
  </head>
<body>
  <h2>Weather Data</h2>
  <p>Random Numbers for Demonstration Purposes</p>
  <table>
    <tr> <th>Parameter</th> <th>Value</th> <th>Units</th> </tr>
    <tr> <td>Temperature</td> <td> aaaaa </td> <td>F</td> </tr>
    <tr> <td>Humidity</td> <td> bbbbb </td> <td>%</td> </tr>
    <tr> <td>Pressure</td> <td> ccccc </td> <td>hPa</td> </tr>
    <tr> <td>Light</td> <td> ddddd </td> <td>Lux</td> </tr>
    </table>
  </body>
</html>
```

Displaying Data: ***web_page()***

- Pass the data to appear in the web page
- Read the text file
- Replace the dummy variables
  - There are probably other and better ways to do this
  - but this works...

```
def web_page(x0, x1, x2, x3):
    f = open("Table.html","rt")
    x = f.read()
    x = x.replace('\r\n',' ')
    x = x.replace('aaaaa', str(x0))
    x = x.replace('bbbbb', str(x1))
    x = x.replace('ccccc', str(x2))
    x = x.replace('ddddd', str(x3))
    return(x)
```

Testing Web Page

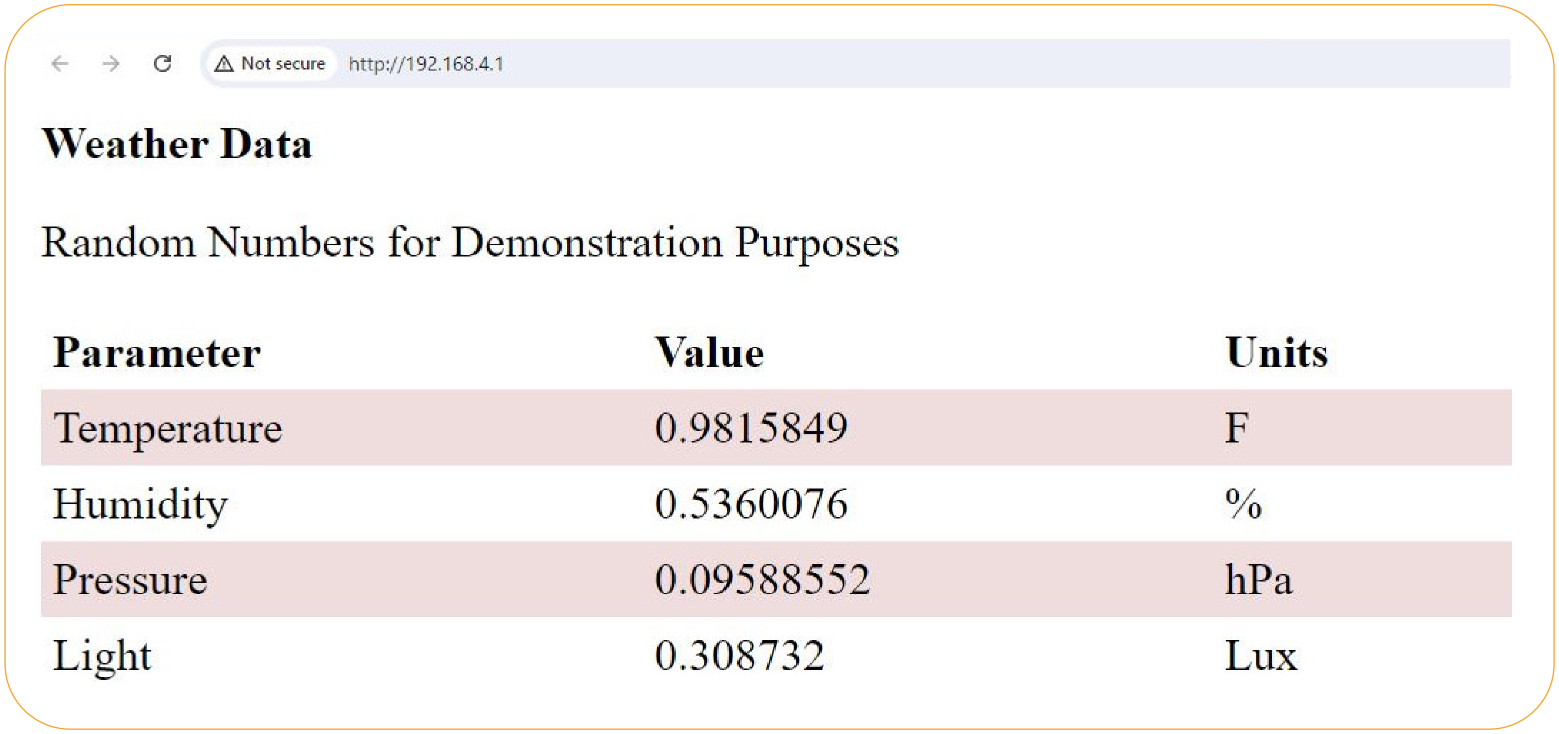
Pass random numbers

Each time you refresh the screen (F5), the data updates

```
while(1):
    conn, addr = s.accept()
    print('Got a connection from %s' % str(addr))
    request = conn.recv(1024)
    print('Content = %s' % str(request))
    response = web_page(random(), random(), random(), random())
    conn.send(response)
    conn.close()
```

Result:

- Data appears in the table
- Each time you refresh (F5),. the data changes



Weather Data

Random Numbers for Demonstration Purposes

Parameter	Value	Units
Temperature	0.9815849	F
Humidity	0.5360076	%
Pressure	0.09588552	hPa
Light	0.308732	Lux

Summary: AP Mode

In AP mode,

- The Pi-Pico W sets up a stand-alone WiFi network
- Other devices can connect to this network as clients

Each ping, the Pico can reply with a web page

- By changing the data in the web page, the clients can see what's going on
- Two-way communications is also possible
  - Next lecture

References

- https://www.youtube.com/watch?v=cZNoXXIEPbg
- https://medium.com/@shilleh/creating-a-wireless-network-with-raspberry-pi-pico-w-part-1-c896211f2bd6
- https://www.w3schools.com/html/default.asp