CS50

Internet Basics

Overview

Programming isn't just limited to writing programs that run on the command line. Code can also be written for web browsers and and shared on the Internet. To be able to share information anywhere, a standard set of rules for sending, receiving, and interpreting the information must be set. Because the Internet connects people and computers from all over the world, there are many different systems and protocols in place that work together in order to allow people to use the Internet effectively. Understanding what these are and how they work will enhance your understanding of the Internet and computer science overall.

Key Terms

- IP address
- access point
- DHCP
- DNS
- URL
- TCP
- HTTP

IP Addresses

Devices on the Internet are assigned an **IP address** (which stands for Internet Protocol) to help identify them and allow them to be found by other devices also on the Internet. IP addresses take the form of `#.#.#.#`, where each `#` is a number in the range of 0 to 255. This allows for about 4 billion possible IP addresses. Although this may sound like a lot, there are far more devices on the Internet than that. This has led to some workarounds, including assigning some devices private IP addresses that together share a single public IP address.

In the long term, however, the 32-bit IP address scheme (IPv4) we mostly use today will be replaced by a 128-bit address scheme called IPv6. While IPv4 addresses take the form of 4 numbers, each representing an 8-bit value, IPv6 addresses have 8 numbers – each representing a 16-bit value – in the form `#:#:#:#:#:#:#:#`.

When information is being sent across the Internet, IP addresses are used so that the Internet knows where the information is being sent to and from. In this sense, it's very much like sending physical mail: information has a both a mailing (to) and a return (from) address.

Connecting to the Internet

Several steps are involved in connecting a device to the Internet. For a wireless device (like a laptop or cell phone), it must first connect wirelessly to an **access point** (known as an AP). For many consumers, this access point takes the form of a home router. This access point is connected to a switch, which is connected to another router, which can then connect to the rest of the Internet.

Two other servers are particularly important for connecting to the Internet: DHCP and DNS. **DHCP**, which stands for Dynamic Host Configuration Protocol, is responsible for assigning computers IP addresses. Early in the Internet Age, network administrators had to manually assign IP addresses to all computers, but thankfully, DHCP now automates this process.

It would be very difficult if everyone using the Internet had to remember each IP address for every website they wanted to visit. Instead, most people type a text-based address (such as "google.com") into their web browsers to access a page. This text-based address is called a **URL**, or Uniform Resource Locator. But how do our computers know what IP addresses to take us to based on our text input?

This is where **DNS**, which stands for Domain Name System, comes in. The term DNS refers to servers that take URLs and converts them to and from IP addresses. When a user types a URL into their web browser, DNS servers look up the URL and then determine which IP address that name refers to, relating this important information back to the computer.

Other Protocols

Several other protocols are involved in ensuring that communication on the Internet works effectively. **TCP**, the Transmission Control Protocol, is responsible for guaranteeing the delivery of all data packets that are submitted via the Internet. It also makes sure that these packets of information sent via the Internet know what service they are meant for (web browsing, email, etc.). **HTTP**, the Hypertext Transfer Protocol, is another protocol which helps web browsers communicate with server.

CS50

IP Addresses

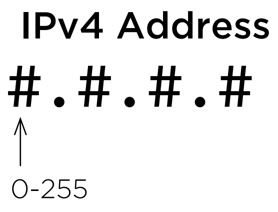
Overview

The **Internet Protocol** is a protocol, or set of rules, that helps define how information on the Internet is transmitted. Part of this protocol involves assigning each device on the Internet an **IP Address**, which helps to identify that device on the Internet. IP has gone through several different versions, the most recent of which is **IPv6**, which is intended to replace the existing protocol, **IPv4**.

Key Terms

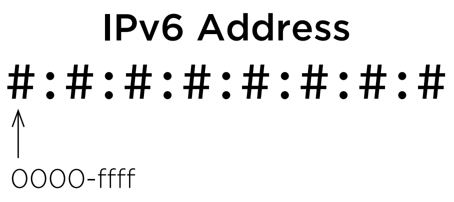
- Internet Protocol
- IP Address
- IPv6
- IPv4

IPv4 and IPv6 Addresses



Under the IPv4 system, each IP address is composed of four numbers separated by decimal points. Each number is a decimal number in the range of 0 to 255 inclusive (8 bits of space). As a result, each IPv4 address is 32 bits, which means there can be at most 232 addresses under IPv4. This amounts to about 4.3 billion addresses total.

However, as the Internet has grown, 4.3 billion addresses is no longer sustainable to support all of the devices that are trying to connect to the Internet. As a result, the IPv6 standard was developed in order to add more possible IP addresses.



Under IPv6, each IP address consists of eight numbers, separated by semicolons. Each number is a 16-bit number (compared to the 8-bit numbers used in IPv4). Instead of representing each number as a decimal, IPv6 uses hexadecimal (16-bit) instead, in the range of 0000 to fffff. Since each IPv6 address stores 128 bits (8 numbers * 16-bits), that means that there are more than 340 billion billion billion billion possible IP addresses. This is significantly more addresses than are currently used, so many IPv6 addresses currently include several 0s among their 8 component numbers.

As a shorthand method, IPv6 addresses can be abbreviated by cutting off any leading 0s in front of hexadecimal numbers and replacing multiple consecutive component 0s with a double colon (`::`).
For instance, the IP address `28aa:0000:0000:0000:0000:0000:0018:a5b2` could be abbreviated to just `28aa::18:a5b2` by removing leading 0s and replacing multiple consecutive 0s with double colons. Note that there can only be one double colon per address in this abbreviated format.

Private IP Addresses

Not all IP addresses are accessible on the Internet at large. Some addresses, known as private IP addresses, are set aside to be used within a particular local network. Other computers on the local network can communicate with one another via their private IP addresses, but computers outside of the network don't have access to them.

Often, devices with private IP addresses will share a single public IP address. This helps reduce the number of public IP addresses that are needed under the IPv4 standard. Certain ranges of IPv4 addresses, such as those which take the form `10.#.#.#`, `172.16.#.#` - `172.31.#.#`, or `192.168.#.#`, are set aside to be used specifically for private IP addresses.

The IP address `127.0.0.1` is an IP address that connects to the same machine that the user is currently using, rather than connecting to a different one. It is known as the loopback address, or the “localhost.” In computers that use IPv6, this address is `0:0:0:0:0:0:0:1`, or `::1`. You know what they say, there’s no place like `127.0.0.1`.

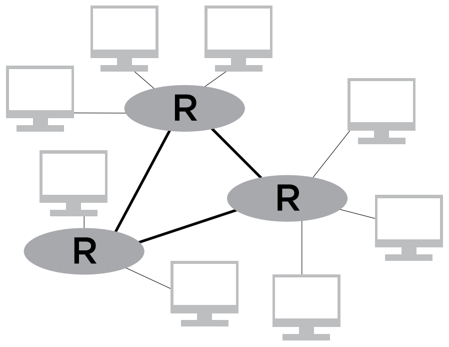
CS50 TCP and IP

Overview

In order for computers to communicate across the Internet, they need a standard set of rules—or **protocols**—to dictate how the communication should happen and how the data should get from one place on the Internet to another. Without these standard ways of communicating information, computers would not be able to guarantee that the receiver would get the information or that the receiving computer would know what to do with it. Two important protocols deal with this: the Transmission Control Protocol—also known as TCP—and the Internet Protocol—or IP. These are often collectively known as TCP/IP.

Key Terms

- protocol
- IP
- router
- IP Address
- packet
- TCP
- port



Internet Protocol

Recall that the Internet Protocol (**IP**) defines how information is transferred from one computer to another. It is structured as a web of connected **routers** (labeled as "R" in the diagram to the left), which are devices that help send information from one computer to another. Data will often need to pass through multiple routers to get from the sender's computer to its destination. Each router is programmed with a set of instructions (stored in a "routing table") that determine the direction in which the data must be sent so that it reaches its final destination.

IP Addresses

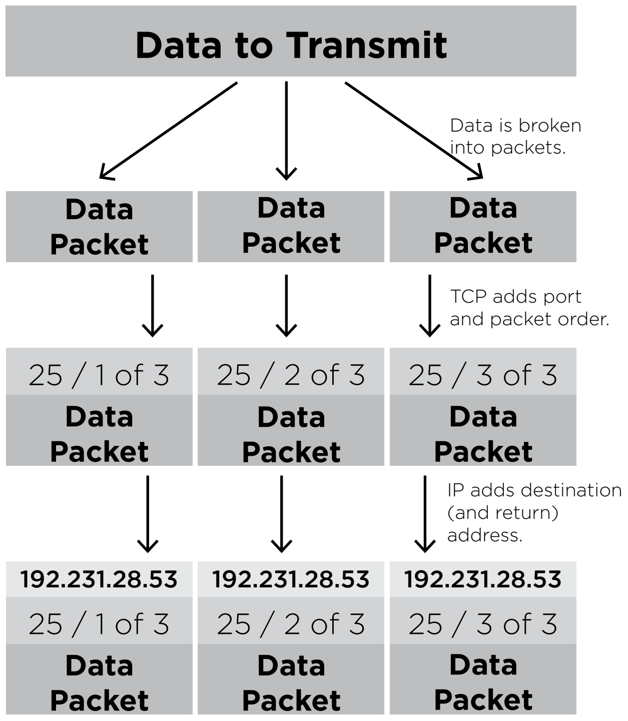
Just as homes need addresses so that mail can be delivered from one house to another, computers need addresses as well so that routers know where information is being sent from and where information should be sent to. These addresses are known as **IP Addresses**, and they take the form `#.#.#.#`, where each `#` stands for a number in the range 0 to 255. When a user types a web address (like google.com) into their web browser, a Domain Name System (DNS) server translates the web address to an IP address (like `172.217.0.46`).

Transmission Control Protocol

Instead of sending all of the data that one computer wants to send to another as one big packet, information on the Internet is sent in smaller data **packets**. The Transmission Control Protocol (**TCP**) is responsible for breaking up data into ordered packets. Since there is no guarantee that data packets will arrive at the destination at the same time, or even in the correct order, TCP labels each packet with the order it should go in. This way, the receiving computer can reassemble the packets in the right order.

TCP can also ask for the retransmission of lost data packets. Additionally, it assigns data a **port** number that indicates what type of internet service the data should be used for. For instance, SMTP (email) uses port 25, while HTTP (normal web browsing) uses port 80.

In summary, to get data across the Internet, TCP first breaks it down into smaller packets. Then TCP labels each packet with a port and packet number, IP tells the packet its destination, and the data is transmitted via routers which eventually direct the packet to its final destination.



CS50 DNS and DHCP

Overview

There are two important systems in place to make sure that devices on the Internet use IP addresses effectively. The first is the Domain Name System, or **DNS**, which is responsible for converting the words that are typed into an address bar in a web browser like Google Chrome or Safari into the corresponding IP address. The second is the Dynamic Host Configuration Protocol, or **DHCP**, which helps assign each device an IP address.

Key Terms

- DNS
- DHCP
- URL
- domain

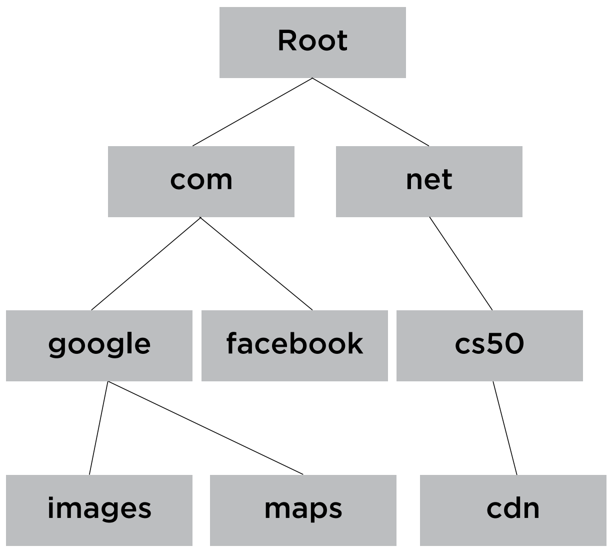
DNS

Most people browsing the Internet don't type an IP address in when they want to access a web page. Instead, they type in a **URL**, a Uniform Resource Locator, which acts as a more human-readable and memorable web address than an IP address.

However, IP still requires the computer to know which IP address it is trying to access. This is where DNS comes in. DNS is responsible for taking the **domain**, which is just an identifier like "google.com" or "facebook.com", and translating it into its respective IP address(es).

When a user types a URL into a web browser, the computer contacts a DNS server, which stores information about which domain names map to which IP addresses. There are many DNS servers, and not all of them will updated at the same time when the mappings between domain names and their IP addresses are changed. As a result, and because it takes time for these changes in the DNS system to propagate throughout all of the DNS servers on the Internet, DNS servers must always communicate with one another about these updates.

DNS Hierarchy



Domains in DNS are organized in a tree-like hierarchy. There are a set of basic "top-level domains" (TLDs), which appear at the ends of many familiar websites. Two hierarchies exist at this level: organizational and geographic. Amongst the top-level organizational domains are com, edu, gov, net, org, among others. The geographic domains are two-letter country codes (such as uk, es, fr, ar).

Website URLs must branch off from one of these top-level domains. For instance, "google.com" branches off of the "com" top-level domain. And “google.co.uk” branches off both the “com” organizational domain and the “uk” geographic domain. Some websites, like "images.google.com" and "maps.google.com", branch even further and are known as subdomains.

DHCP

Computers, and the humans that use them, need a system for allocating these IP addresses. At one point in the Internet's history, a human network administrator was responsible for this, assigning IP addresses to computers manually. Nowadays, the Dynamic Host Configuration Protocol, or DHCP, can do this automatically. When computers connect to a network, they will connect to a DHCP server. The DHCP server then accesses a pool of available IP addresses and assigns each computer on the network a unique one.

So, using DHCP and DNS, devices on the Internet are able to receive their own IP address and determine which IP address corresponds to the website that a user is trying to visit. These systems are crucial, allowing the Internet Protocol to effectively facilitate communication across the Internet.

CS50 HTTP

Overview

The Hypertext Transfer Protocol, or **HTTP**, is a protocol for how web browsers communicate with web servers. When a user wishes to visit a webpage, their web browser (which may be referred to as the **client**) must request the contents of the web page from a web **server**. In response, the web server must interpret the request and send the requested page back to the client. HTTP facilitates this process and sets a standard way for these requests to be sent and received.

Key Terms

- HTTP
- client
- server
- GET
- POST
- Status Code

GET and POST Requests

```
GET / HTTP/1.1
Host: www.google.com
```

When a user requests a web page by typing a URL into their web browser, the web browser sends a particular type of HTTP request called a **GET** Request. The text of a GET request begins with the word `GET`, to indicate the request type. Following the word `GET` is a path indicating which web page the user is requesting, called the "Request URI," where URI stands for Uniform Resource Identifier.

Following `GET` is `/`, which indicates the root of the web page, such as when you type a URL like `google.com/` or `facebook.com/` without specifying anything after the `/`. Finally, the first line of the GET request will end with the version of the HTTP protocol that the request is using, generally `1.1`. The next line specifies the "Host," which is the domain name which the user is requesting a page from.

Web browsers can also submit a different type of HTTP request, called a **POST** request, which is meant for transmitting data from the client to the web server, such as when a user fills out an online form. In this case, a client would have asked the server for the blank form via a GET request. Once submitted, the filled out form would be sent back to the server via a POST request.

Status Codes

When a web server receives an HTTP request from a client, the server must send a response back to the client. Servers indicate the results of requests with **status codes**, which they send back to the client.

For instance, if a client requests a web page, the server should send back the contents of that web page. If the server has the page that the user is looking for and is able to respond with it successfully, then the server sends the status code `200`, which means that the request was handled successfully. But if the user requests a page that doesn't exist on the web server, then the server responds with status code `404`, which stands for "Not Found."

Other types of errors are also represented by status codes. For example, if the user tries to access a web page that the user does not have permission to access, then a web server will respond with status code `403`, which means "Forbidden." If an error occurs in the web server while trying to process a user's request, then the server will frequently respond with status code `500`, which stands for an "Internal Server Error."

```
HTTP/1.1 200 OK
Content-Type: text/html
```

Status Code	Meaning
200	OK
301	Moved Permanently
302	Found
403	Forbidden
404	Not Found
500	Internal Server Error

CS50

HTML

Overview

HyperText Markup Language, or **HTML**, is the language that describes the contents of web pages. It's not a programming language (it doesn't contain loops, if statements, etc.), but it does allow a web designer to decide how a web page is laid out. When a user requests a web page, the web server will send the contents of that page in HTML. The web browser interprets this HTML and displays the web page to the user.

Key Terms

- HTML
- element
- tag
- DOM
- attribute

HTML Basics

To the right is a basic, sample HTML web page. The first line clarifies that the document is an HTML document (specifically, an HTML5 document). After that, HTML code is organized as a series of nested **elements**. Each element begins and ends with a **tag** that indicates what type of element it is. Anything between those tags is the contents of the element. In our example, the outermost element is the html element, since it is enclosed with `<html>` at the start, and `</html>` at the very end. Everything in between those two tags is part of the html element.

```
<!DOCTYPE html>

<html>
    <head>
        <title>hello, world</title>
    </head>
    <body>
        Hello!
    </body>
</html>
```

In general, opening tags will take the form `<tagname>` and closing tags will take the form `</tagname>`. Inside of our `html` element are two other elements: `head` and `body`. The `head` element contains information about the webpage that isn't in the body of the web page itself. For instance, the head element here contains a `title` element, which sets the title of the webpage to `hello, world`. Inside of the body element, which defines what's actually in the main content area of the webpage, is just the word `Hello!`

You can think of the structure of an HTML document as a tree-like hierarchy. The `html` element is at the root of the tree. Branching off of it are the head and body tags. Branching off of the `head` tag is the `title` tag, and so forth. This model of viewing an HTML document as a tree is known as the Document Object Model, or **DOM**.

Common Tags

HTML offers many tags that can be used to format a web page. The ones in our example – `html`, `head`, `title`, and `body` – are very common and will likely appear in every web page we write. Other tags may appear in particular situations. For example, headings in web pages are denoted by the tags `<h1>` through `<h6>`, where `<h1>` is the largest, main heading, and each subsequent one is smaller. The `<p>` tag denotes a paragraph.

Element Attributes

In addition to having a name, HTML elements can also have **attributes**, which provide additional information about them. For instance, the `<img>` (image) tag takes an attribute called `src`, which specifies the address for, or where to find, an image. So a tag like `<img src="cat.jpg">` would place the image cat.jpg on the webpage, so long as `cat.jpg` is in the same directory as the HTML document.

To create links to other pages in HTML, we use the `<a>` tag. The `<a>` tag takes an attribute called `href`, which specifies what web page the link should link to. As a result, HTML like `<a href="http://google.com">Click Here</a>` would create a link labeled "Click Here" which, when clicked, would take the user to Google.

All HTML elements can also have an `id` attribute, which must be unique. The `id` can help to identify particular elements in the webpage. HTML elements can also have a `class` attribute, which does not have to be unique and can also be used to identify HTML elements. We'll see the utility of the `id` and `class` attributes when we begin dealing more with CSS and JavaScript.

CS50

CSS

Overview

Cascading Style Sheets (**CSS**) is a language used on the Internet to style web pages. While HTML describes the structure of a web page, CSS determines text alignment, the size of various elements, the color of various elements, and how HTML elements appear as a window is resized, amongst other things. There are also several different ways of incorporating CSS into a web page.

Key Terms

- CSS
- attribute-value
- ID
- class

The Style Attribute

CSS can be included directly into HTML using the style HTML attribute in any HTML tag. CSS takes the format of **attribute-value** pairs, where each CSS attribute is followed by a colon, followed by its value (multiple attribute-value pairs can be separated by semicolons). In the example here, we've included CSS directly in this `<p>` tag. The CSS attribute font-size is set to `28px`. As a result, when the HTML is displayed in a web browser, the paragraph will appear in 28-point font.

```
<p style="font-size:28px;">
    This is a paragraph.
</p>
```

Note that there is a distinction between HTML attributes and CSS attributes. `style` is the name of an HTML attribute, while `font-size` is an example of a CSS attribute. There are many different CSS tags. Common ones for styling text include: `color`, which sets the color of the text; `text-align`, which sets the text alignment (centered or left-aligned, for example); and `font`, which sets the font for the text.

```
<!DOCTYPE html>
<html>
  <head>
    <title>Page</title>
    <style>
        p
        {
           font-size: 28px;
        }
    </style>
  </head>
  <body>
    <p>This is a paragraph.</p>
  </body>
</html>
```

The Style Tag

CSS can also be located inside of a `style` element, usually located in the head section of an HTML document. Within the `style` tags, we first need to specify what the styling should apply to. This could be the name of a type of element (e.g. `p`), or it could be an ID or class of an HTML element. When determining whether to use an ID or a class, you can think of how we use the words in an everyday context. An **ID** is typically unique to a user so it should be used to style only one element. A **class** meanwhile is a group of students so it should be used to style multiple elements that have something in common. To apply styling to an ID, the ID should be referenced with a `#` symbol followed by its name. To apply styling to a class, the ID should be referenced with a `.` followed by the name of that class.

After specifying what their styling should apply to, CSS attribute-value pairs can be included within curly braces, separated by semicolons. In the example at left, the CSS specifies that all `p` elements should have font size `28`. Styling CSS in this manner can be advantageous if the styling applies to multiple different HTML elements, since then we don't repeat the same CSS.

Factored CSS

A third way to style with CSS is to store all the CSS in an entirely separate file. If the all of the CSS that would be inside of `<style>` tags is stored in a document (typically one that ends with `.css`), then that file can just be imported into an HTML document. We can do this by including a `<link>` tag in the head element of the HTML document. For instance, by including a line like `<link href="style.css" rel="stylesheet" />`, we can add an external CSS document to the HTML document and know that the appropriate styling will appear.

By putting all of the overlapping CSS code into one file, we remove unneeded repetition and redundancy from our HTML files. We also make changing the styling of multiple HTML documents more efficient since we would only need to update the one CSS document and then those HTML documents would be automatically re-styled. In these ways, factoring out CSS into a separate document can be particularly advantageous when dealing with multiple different HTML documents that use the same styling.

CS50 JavaScript

Overview

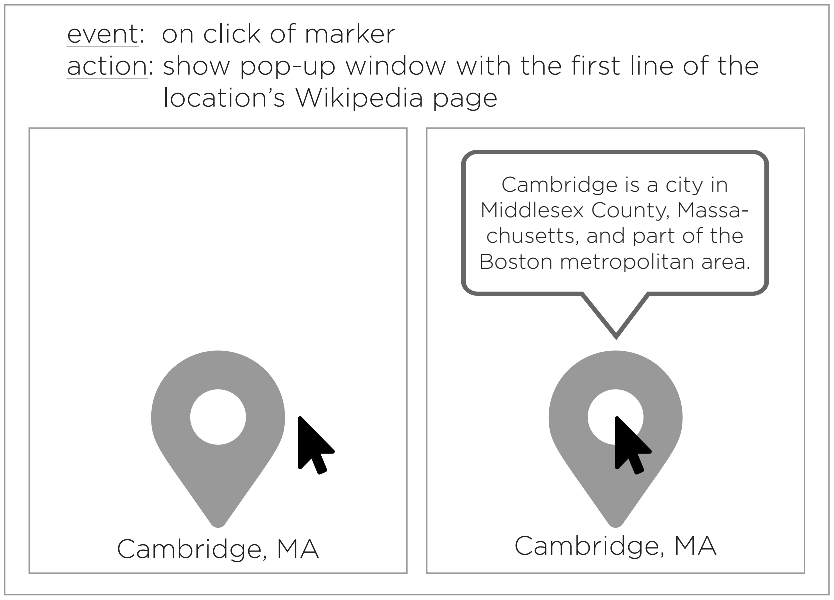
If you've seen websites with cool animations when you interact with them, it's likely that those features were written in JavaScript. Each programming language plays a certain role in computer science, and JavaScript's job is in the web browser. JavaScript is a language used in web development to program the behavior of web pages. Because JavaScript was created for this use, JavaScript has many features that other languages such as Python, C, and Java do not have.

Key Terms
- JavaScript
- event
- anonymous function
- jQuery

JavaScript

JavaScript is a programming language used in web browsers to create dynamic web pages. Because JavaScript runs in all major web browsers, it is universally supported and used in the majority of modern websites. JavaScript is primarily used on the client-side of web applications, meaning the code is run on a user's browser rather than somewhere on a external server. This means that without additional HTML requests, JavaScript does not have access to information on the server.

JavaScript works closely with HTML and CSS to create user interfaces for web applications. Javascript is written within script tags of an HTML page and has many functions that allow for direct manipulation of HTML using the DOM. Part of what makes JavaScript so powerful and dynamic is its focus on responding to **events**, or actions from a user or other devices. An event-driven language is one that enables a program to act according to these events. Some examples of events include mouse clicks, key presses, scrolling, or outputs from sensors. The job of many JavaScript programs is to do something depending upon what the user does, and JavaScript provides built-in features to support this kind of functionality.


event: on click of marker
action: show pop-up window with the first line of the location's Wikipedia page
Cambridge is a city in Middlesex County, Massachusetts, and part of the Boston metropolitan area.
Cambridge, MA
Cambridge, MA

One of these features is unnamed functions, or **anonymous functions**. Functions are named so we can call them in various parts of our program, but since JavaScript is event-driven, many actions are only triggered by particular events. In other words, it is common to have functions that are only used once in a program. Therefore, JavaScript allows us to create anonymous functions. Unlike declared functions, anonymous functions run immediately without storing them in memory first. This simplifies code, lets us assign functions as variables, and allows us to pass functions around as arguments in other functions. All of these features simply make creating interactive and user-friendly web pages more convenient.

Additional Resources

JavaScript is widely used among web developers, so there are many JavaScript resources available to make coding more convenient and to expand the range of possibilities. One of these resources is a library called jQuery. **jQuery** is a cross-platform JavaScript library that simplifies some of JavaScript's syntax and includes some additional helpful functions. It uses the $ symbol as a global variable to access jQuery. In addition to jQuery, JavaScript has a plethora of resources available. Some of these resources include libraries of pre-built JavaScript components and tools for data visualization. Because web development brings together so many technologies and has such a big impact on how information is presented, there are constantly new resources being developed around JavaScript. Part of web development is keeping up with these technologies and being able to adapt quickly. Be sure to check out which resources could be useful for your next project!