The Internet

CS106AP Lecture 27

Lecture examples and slides used with the permission of Brahm Capoor
Today’s questions

How does the Internet work?
Today’s topics

1. Review
2. The Internet
3. Clients and Servers
4. Analyzing online data
5. What’s next?
Review
GImage
GImages

- Like SimpleImages but as GObjects!

```python
image.height, image.width
image.get_pixel(x, y)
image.set_pixel(x, y, pixel)
image.x
image.y
image.move(dx, dy)
```

- SimpleImage functions!
- GObject functions!
GlImages

- Like SimpleImages but as GObjects!

- The differences
  - Declaration
    ```python
    image = GImage.from_file(filename_string)
    ```
  - No image resizing or creation of blank images
  - Graphics **Pixel** objects are not associated with the image
How to loop over a \texttt{GImage}

You must call \texttt{.set\_pixel()} to update the image after editing the pixel!

\begin{verbatim}
for y in range(image.height):
    for x in range(image.width):
        # Use \texttt{pixel = image.get\_pixel(x, y)} to get pixel
        # Edit \texttt{pixel.red, pixel.green, pixel.blue}
        # Use \texttt{image.set\_pixel(x, y, pixel)} to update pixel
        # within the image
\end{verbatim}
How your computer works
1. Storage
2. Central processing unit (CPU)
3. Random Access Memory (RAM)

Users (us!)
Running a program

● Your code file is stored in storage (persistent memory).

● The CPU runs your program and is powered by one or multiple cores.

● When you “run” programs (or applications), each program gets its own space in RAM.

● When the program exits, the space it took in RAM gets reclaimed.
  ○ Who keeps all this organized in RAM? Your operating system (OS)!
The operating system

- It’s the first program that gets run when you turn on your computer.
- It stops and starts other programs and manages the space they take up in RAM.
- It’s what we’re talking to when we type commands into Terminal/the command line!
Binary
Binary numbers

- Internally, computers represent numbers using a number system called binary
  - Each binary digit is called a bit and can take on a value of 0 or 1
  - This matches up well with the fact that your computer’s hardware is composed of many different switches, each of which can be “on” (1) or “off” (0)

- The number of bits used to store a piece of information is directly related to how many different values that piece of information can take on
  - Ex: Each color channel in an RGB pixel is represented using 8 bits, which means that there are $2^8 = 256$ possible different values that a color channel can take on

- Data storage capability of computers is usually represented in terms of bytes, where a byte is defined as 8 bits of information
  - Ex: 8 Gigabytes of RAM means roughly 8 billion bytes (or 64 billion bits) of storage space
How does the internet work?
Yesterday’s Lecture
Today’s Lecture
What is the Internet?
What is the Internet?
What is the Internet?
What is the Internet?

6AM ALARM RINGS
ME:

2 HOURS LATER
STILL ME:

Facebook

Wi-Fi
What is the Internet?
The Internet

a global computer network consisting of interconnected computers using standardized communication protocols.
The Internet

a global computer network consisting of interconnected computers using standardized communication protocols.

Definition

Many computers talking to one another
The Internet

a global computer network consisting of interconnected computers using standardized communication protocols.
Clients and Servers
The Internet

In simple terms, the Internet consists of many programs running on many computers sending messages to one another (usually as strings)
The Internet

In simple terms, the Internet consists of many programs running on many computers sending messages to one another (usually as strings)
The Internet

In simple terms, the Internet consists of many programs running on many computers sending messages to one another (usually as strings).
The Internet

In simple terms, the Internet consists of many programs running on many computers sending messages to one another (usually as strings).

Facebook datacenter

Your computer (facebook.com)

“Server”

“Client”
The Internet

In simple terms, the Internet consists of many programs running on many computers sending messages to one another (usually as strings)

Facebook datacenter

Your computer (facebook.com)

“Server”

“Client”

Get status for “Nick Bowman”

request
The Internet

In simple terms, the Internet consists of many programs running on many computers sending messages to one another (usually as strings).
The Internet (the gritty details)

- Your browser ("client") sends a **HTTP** GET request to some other computer running code ("server")
  - HTTP is the shared communication protocol that allows computers all around the world to talk to one another
  - A GET request does exactly that the name implies— it tries to get some information
The Internet (the gritty details)

- Your browser ("client") sends a HTTP GET request to some other computer running code ("server")
  - HTTP is the shared communication protocol that allows computers all around the world to talk to one another
  - A GET request does exactly that the name implies— it tries to get some information

- The server replies with some response data in the form of HTML
  - HTML is the standard language used to display content in web browsers
  - If you want to see some HTML, visit any webpage, right-click and select the "View Source" option
  - HTML defines every web page you’ve ever visited!
The Internet (the gritty details)

- Your browser (“client”) sends a HTTP GET request to some other computer running code (“server”)
  - HTTP is the shared communication protocol that allows computers all around the world to talk to one another
  - A GET request does exactly that the name implies— it tries to get some information
- The server replies with some response data in the form of HTML
  - HTML is the standard language used to display content in web browsers
  - If you want to see some HTML, visit any webpage, right-click and select the “View Source” option
  - HTML defines every web page you’ve ever visited!
- Many clients can connect to the same server
  - Ex: Facebook datacenter near Bend, Oregon
Accessing the Internet in your own code

- Python’s `urllib` module allows us to request information like text or HTML files from the internet like so (must import `urllib.request`):
Accessing the Internet in your own code

- Python’s `urllib` module allows us to request information like text or HTML files from the internet like so (must import `urllib.request`):

```python
>>> f = urllib.request.urlopen('http://web.stanford.edu/class/cs106ap/hello.txt')
```
Accessing the Internet in your own code

- Python’s `urllib` module allows us to request information like text or HTML files from the internet like so (must import `urllib.request`):

```python
>>> f =
urllib.request.urlopen('http://web.stanford.edu/class/cs106ap/hello.txt')

# Read the bytes of the request and interpret them as a string

>>> f.read().decode('utf-8')

'Hello from CS106AP url!
What if there were data here?
12,34
25,19
66,0
1,2
'
```
Accessing the Internet in your own code

- Python’s `urllib` module allows us to request information like text or HTML files from the internet like so (must import `urllib.request`):

```python
>>> f = urllib.request.urlopen('http://web.stanford.edu/class/cs106ap/hello.txt')
# Read the bytes of the request and interpret them as a string
>>> f.read().decode('utf-8')
'Hello from CS106AP url!
What if there were data here?
12,34
25,19
66,0
1,2
'
```

- Another example of a powerful Python library!
Online hello.txt example
[interpreter demo]
<not a meme>
<<this is real life>>
Jessie Salas is at Vaden Health Center.
July 22 at 3:10 PM · Stanford, CA

<not a meme>
<< this is real life >>
Building a chat application

Even the world’s simplest chat application must have two kinds of request:

- **Getting** all the messages from a server
- **Sending** a message to a server

and any chat server should gracefully deal with both requests somehow
Chat Application

[code demo]
If you’d like to try the chat app, download the lecture code!

The logfile for the app can be found here.
Real-World Chat Applications

Now, we have some sense of the kinds of things that a chat application needs to do in order to function.

More importantly, we’ve seen that servers can store and log all of the messages that are sent.
Real-World Chat Applications

Now, we have some sense of the kinds of things that a chat application needs to do in order to function.

More importantly, we’ve seen that servers can store and log all of the messages that are sent.

Let’s do something cool...
Download your information

You can download a copy of your Facebook information at any time. You can download all of it at once, or you can select only the types of information and date ranges that you want. You can choose to receive your information in an HTML format that is easy to view, or a JSON format, which could allow another service to more easily import it.

Downloading your information is a password-protected process that only you will have access to. Once you’ve created a file, it will be available for download for a few days.

If you’d like to view your information without downloading it, you can Access your information at any time.

[Options for selecting data to download]

facebook.com/dyi
The first Facebook message I ever received:

did u get the lost hero for xmas?
The first Facebook message I ever received:

*did u get the lost hero for xmas?*
The first Facebook message I ever received:

did u get the lost hero for xmas?

Since then, I have sent or received close to 419,000 messages. Today, we’re going to explore some of the secrets hidden in these messages.
Each line of the file looks like this:

\(<\text{timestamp}\>\ \ <\text{friend}\>\ \ <\text{message}\>\ \ <\text{status}\>

\(<\text{timestamp}\>): how many months after my account’s creation the message was sent

\(<\text{friend}\>): my conversation partner

\(<\text{message}\>): the body of the message

\(<\text{status}\>): Whether I received or sent the message.

These fields are separated by tab characters (‘\(\text{t}\)’).
Getting your own data

- You can download your own data from facebook, but unfortunately, it doesn’t come in the format we’re using today
  - It’s a format that’s useful for many kinds of programs, but a little too complex for our purposes
- We have provided a Python program that takes the files you can download from facebook, and converts them to the format I described in the previous slide
- If you’re interested in making your own data file, follow the instructions in the next few slides. Otherwise, skip forward to the main lecture.
Getting your own data

- First, request to download your own data from facebook.
  - Click ‘download your information’
  - In the top bar, select ‘JSON’ from the ‘Format’ dropdown
  - Click ‘Deselect All’ at the top right of the options menu, and then scroll down and reselect ‘Messages’
    - You can download whatever else you want (it’s all super interesting!), but the download will take longer to process. We don’t recommend downloading your photos and videos, because the size of the download increases almost exponentially.
  - Click ‘Create File’
  - Facebook will take a while to create the file, but will shoot you an email within about an hour (if you’re downloading only your messages) with a link to the download
    - As a side note, it feels really cool to be doing something that takes even a company the size of facebook a while :)
Getting your own data

- Once you’ve downloaded your data, unzip it and open it up.
- You should see a folder called `messages/` which contains at least some of the following subfolders:
  - `archived_threads/`
  - `filtered_threads/`
  - `inbox/`
  - `message_requests/`
  - `stickers_used/`
- Open the `inbox/` folder, and download and copy this Python program into it. You’ll need to right click the link on this slide and click ‘save as’.
Getting your own data

- Open `parse_messages.py`
- You are not responsible for understanding this code, but it is exhaustively commented if you’re interested in reading it!
- Modify the `MY_NAME`, `ACCOUNT_CREATION_YEAR` and `ACCOUNT_CREATION_MONTH` constants to be your details.
- Open the `inbox/` folder in your terminal, and run this command (it’ll take a while):
  
  ```
  python3 parse_messages.py
  ```

- Feel free to contact us if something appears to go wrong with this script!
Getting your own data

- If `parse_messages.py` completed successfully, a new file should have been created in the `inbox/` folder called `message_logs.txt`
- Copy `message_logs.txt` into today’s lecture code folder, and you’re ready to start analyzing! `chat_utils.py` and `Chat Analysis.ipynb` should already be set up to work with it.
Interrogating the dataset

We’re going to ask two questions of our data today:

1. **Who** do I talk to, and **how much** do I talk to them?
2. **What** do I talk about?
Interrogating the dataset

We’re going to ask two questions of our data today:

1. **Who** do I talk to, and **how much** do I talk to them?
2. **What** do I talk about?

To answer them, we’re going to calculate and store two kinds of data:

1. How often I talk to **each of my friends**
2. How often I use **particular words**
How often do I talk to my friends?

I want to know — for each friend I’ve sent a message to — how many messages I’ve sent them per month that I have messaged them.
How often do I talk to my friends?

I want to know — for each friend I’ve sent a message to — how many messages I’ve sent them per month that I have messaged them.

I want to associate my friend’s names with dates (in this case, months), which are then subsequently associated with chat counts.
How often do I talk to my friends?

I want to know — for each friend I’ve sent a message to — how many messages I’ve sent them per month that I have messaged them.

I want to associate my friend’s names with dates (in this case, months), which are then subsequently associated with chat counts.

Sound familiar? It’s basically BabyNames!
Parsing FB Message Data

[code demo]
How often do I use particular words?

I want to know — for each word that I’ve used in a message — how many messages per month have used that word.
How often do I use particular words?

I want to know — for each word that I’ve used in a message — how many messages per month have used that word.

I want to associate each word with dates (in this case, months) and subsequently with frequencies.
How often do I use particular words?

I want to know — for each word that I’ve used in a message — how many messages per month have used that word.

I want to associate each word with dates (in this case, months) and subsequently with frequencies

Sound familiar? It’s literally exactly what we just did
Parsing and graphing FB Message Data

[code and Jupyter notebook demo]
What’s next?