INFSCI 2560: Web Technologies & Standards

Week ONE

Today's Agenda

Course Overview & Expectations

The Web Overview

- History
- Web Standards
- Web Applications
- HTTP
- Basic Networking Concepts

About Me

- Tonya Edmonds
  - Adjunct Professor, School of Computing and Information
  - MS, Computer Science, University of Pittsburgh
  - Software Engineer > Product Manager
- Interests
  - Intelligent Tutoring Systems
  - Computer Science Education
  - Running
  - Spending time with my family

You can call me Mrs. Edmonds or Professor Edmonds

Tell Me About You

PollEv.com/tonyaedmonds931

Or text **TONYAEDMONDS931** to **37607**

Course Overview

INFSCI 2560

When: Mondays, 6-8:50p

Location: IS 404

Teaching Assistant: Rui Meng (rui.meng@pitt.edu)

Typical Class

6:00 - 7:00	Lecture, part 1
7:00 - 7:10	BREAK
7:10 - 7:40	Lecture, part 2
7:40 - 7:45	BREAK / transition
7:45 - 8:50	In-class activity/discussion

Course Expectations

How to succeed in this course

- Prerequisite Knowledge
  - Some basic familiarity with web technologies (HTML)
  - Programming experience (doesn't matter what language)
  - Ability to learn independently
- Engagement
  - Experiment
  - Fail
  - Plan
  - Collaborate

Course Expectations

- Come to class
- Ask questions
- Take notes
- Be Prepared
  - Read the readings before class
- Do the work
- Bring a laptop to class

Resources

- Course Website: https://glitch.com/~infsci-2560-edmonds
  - The website is the syllabus
  - Will always contain the most recent information
  - Announcements, readings and assignments will be posted here
- CourseWeb
  - For submitting activity deliverables and assignments
- Readings
  - No specific textbook, but I have a list of suggestions on the course website
  - Will also post links to other web resources
  - Mozilla Developer Network and W3Schools
- Lectures
  - Will cover (some) of the readings and other subjects
  - Will also include activities to exercise what you are learning

By the end of this course you will ....

Work independently and on a team to create a website with dynamic content.

Describe both the major technologies and design approach you used to implement your web applications.

Learn about common web application architecture and the technologies behind them.

Develop major component of web sites and applications.

By the end of this course you will ...

Compare and contrast your technology approaches and decisions with other options.

Understand and follow trends in the development of web technologies and standards.

Learn about other web technologies and solve problems on your own.

Course Policies

- Classes will be divided into lecture and activities
- Activities will be structured, hands-on problems related to the lecture
- There will be readings posted on the website each week. You will be responsible for reading them before class.
- Absences: If you miss class you will be responsible for completing the activity before the deadline (on CourseWeb)
- **Plagiarism will not be tolerated - Cite your work!**
  - **For projects, attribute code to the original author and indicate your contribution**
  - **A grade of 0 will be assigned to plagiarised work**

`* Read the course syllabus for the full list of course policies.`

Communication

- Please ask me questions at any time during the lecture.
- During class activities I will be available to help guide you through any difficulties you have.
- Feel free to contact me via email to ask questions or set up a meeting.
- Office hours are by appointment. I am typically available right before class as well.
- If you email me, please make sure to add the course number (INFSCI2560) in the subject line.

Grading

20%	**Exams** - 2 exams - Format: multiple choice & short answer
20%	**Activities** - Each class will have a short, hands-on activity - Due the next day by 11:59p
30%	**Assignments** - Completed individually - Late assignments, 10% off grade for each day, up to 3 days
30%	**Final Project** - Completed in teams of 3-4 - Cannot be submitted late

Activities and assignments will be submitted via CourseWeb.

Activity Rubric

3 points - meets or exceeds all listed criteria

2 points - does not meet all listed criteria

1 point - meets minimal criteria, not complete

Final Project

- Groups of 3-4
- Build a complete web application utilizing the technologies and standards covered in this class
- Each team member will have a primary role (e.g. front-end, back-end)
- Final report (documentation)
- Final presentation

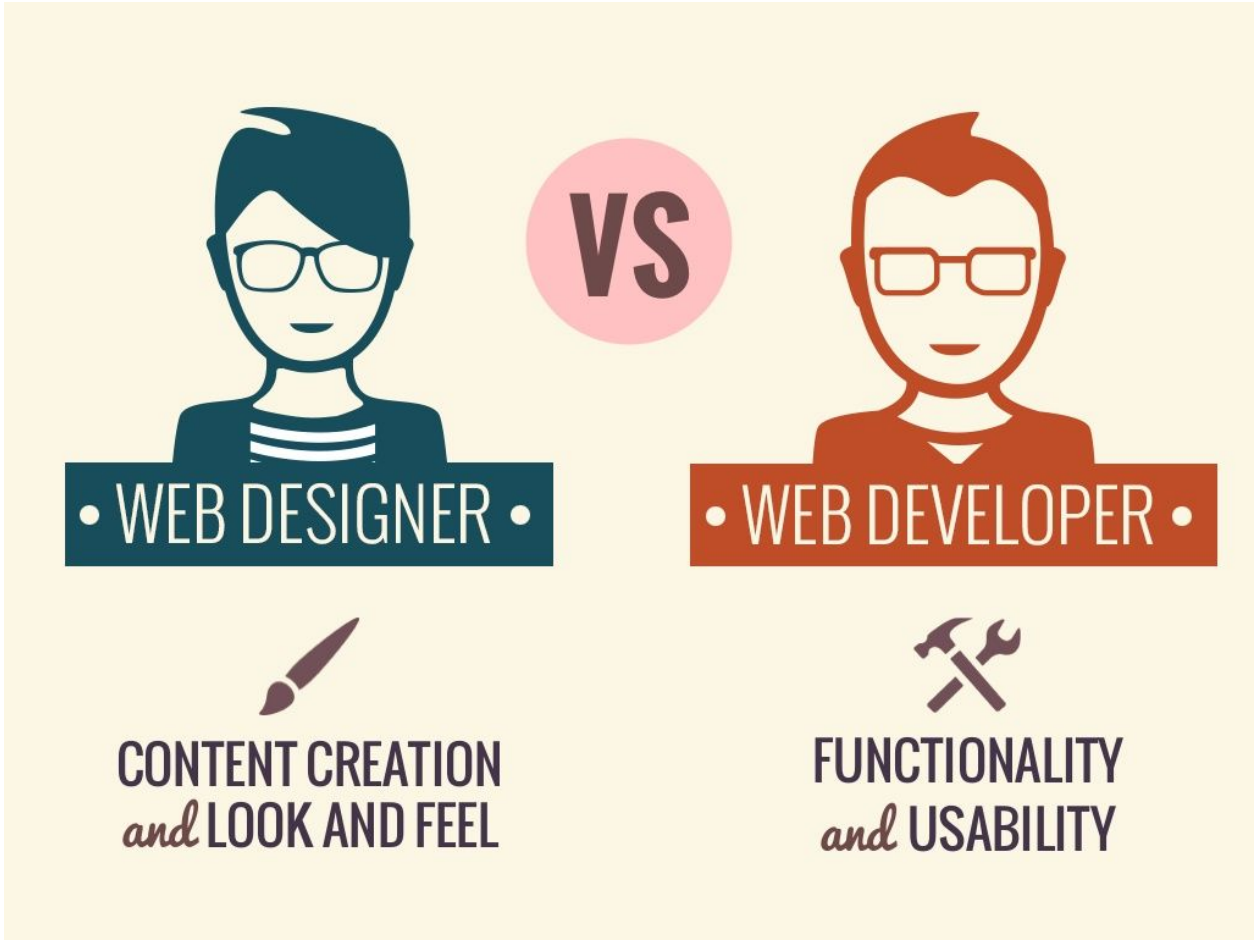
Questions?

DISCLAIMER

Everything mentioned is the intended PLAN for this course. The syllabus may change based on a variety of factors (timing, weather, interest, etc.).

Always check the course website for announcements.

Web Development Roles


VS
• WEB DESIGNER •
• WEB DEVELOPER •
CONTENT CREATION
and LOOK AND FEEL
FUNCTIONALITY
and USABILITY

Front-End Designer

- Focus on the user experience and interactions
- Designs the look and feel of a website (or web application)
- Researches different designs
- Designs prototypes to present to the development team (or client)
- Designs for all devices and platforms
- Designs user flows
- Works with the developer to implement the design

Front-End Developer

- Focuses on coding, not the design
- Creates the inner workings of the website
- Deeper knowledge of the technologies
- Tests browser compatibility
- Uses HTML, CSS, JavaScript, Angular, Bootstrap, etc.

Backend Developer

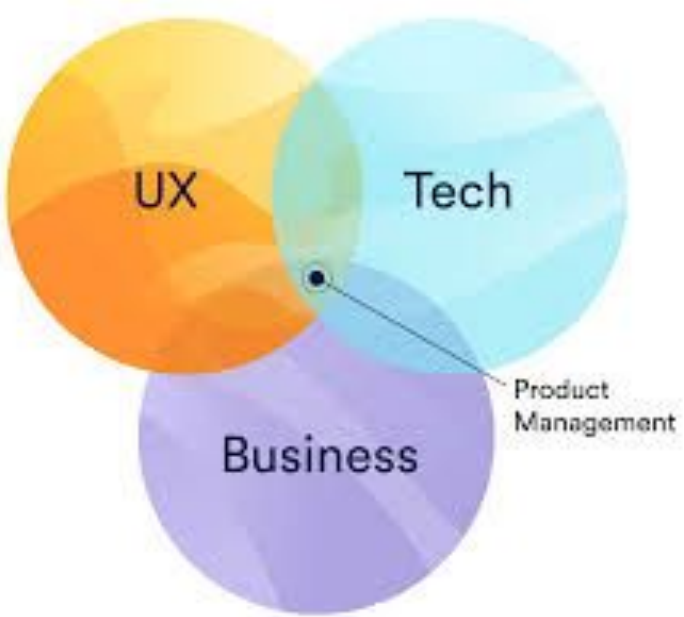
- Responsible for the logic of the website (or application)
- Is left-brained: logic, linear thinking, technical
- Hands-on coding experience is required
- Enables the front-end experience
- Strong technical background
- Knowledge of databases and data storage technologies
- PHP, Java, Ruby, Python, MySQL, NodeJS, MongoDB

Full-Stack Developer

- Comfortable working with both back-end and front-end technologies.
- Works with:
  - Databases
  - PHP/Python/.Net/Ruby/Java
  - HTML
  - CSS
  - JavaScript
  - and everything in between

Product Manager

- Accountable for the success of a product throughout its life cycle
- Defines the product strategy and roadmap
- Must understand the technology
- Works closely with the UX team to define the product
- Works with dev/engineering to execute
- Works with development, marketing, sales, support, etc.

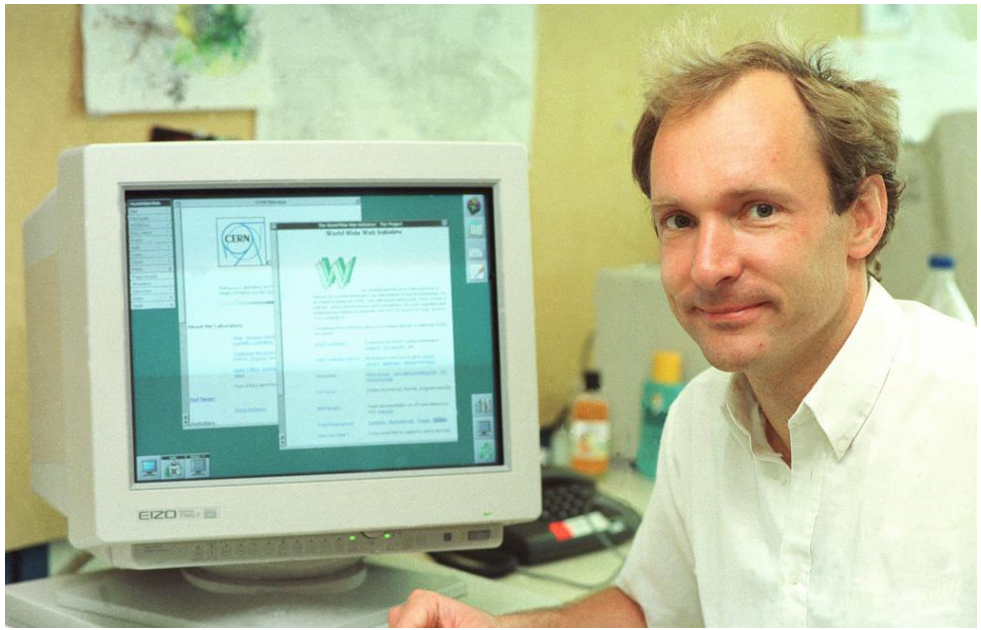


Which role most interests you?

Today's Guiding Questions

1. What is the world wide web?
2. What is the Internet?
3. What is a web application?
4. How is a web application different from a website?
5. What is the role of web standards? Why do we have them?
6. What is HTTP?

Brief History of the Web



- Created in 1989 by Sir Tim Berners-Lee while working at CERN
  - HTML Browser & Editor
  - HTTP
  - WWW Server
- A tool for document management and storage
  - http://info.cern.ch/hypertext/WWW/TheProject.html
- SUPER POPULAR by 1993
  - Mosaic Browser for Mac & Windows

World Wide Web Consortium (W3C)

- Founded by Tim Berners-Lee in 1993
- An international standards organization focused on web technology standards
  - *“Developing protocols and guidelines that ensure long-term growth for the Web.”*
- Is composed of businesses, non-profits, universities, governments, and individuals
- Created XHTML spec

The Role of Standards

- To prevent a fragmented, incompatible ecosystem
  - Or a monolithic ecosystem, i.e. IE
- Create a forum where many diverse stakeholders can come together
  - Open Stand Principles - Due process, Broad consensus, Transparency, Balance, Openness
- Who uses standards?
  - Web developers and designers
  - Web technology implementations (Mozilla, Google, Apple, etc.)
  - You!
- What does a standard look like?
  - HTML 5.2 https://www.w3.org/TR/html52/

The WHATWG and Web Standard Wars

- **Web Hypertext Application Technology Working Group**
- Formed in 2004 by Apple, Mozilla, and Opera
- In response to slow pace of standards development of the W3C
  - And the increased focus on XML technologies, namely XHTML
- Developed the HTML5 and Web Applications 1.0 Standard
  - Increased focus on an application centric vs. document centric web
- HTML 5, Web APIs, DOM are now W3C Standards
  - XHTML is dead, long live XHTML
- Who won?

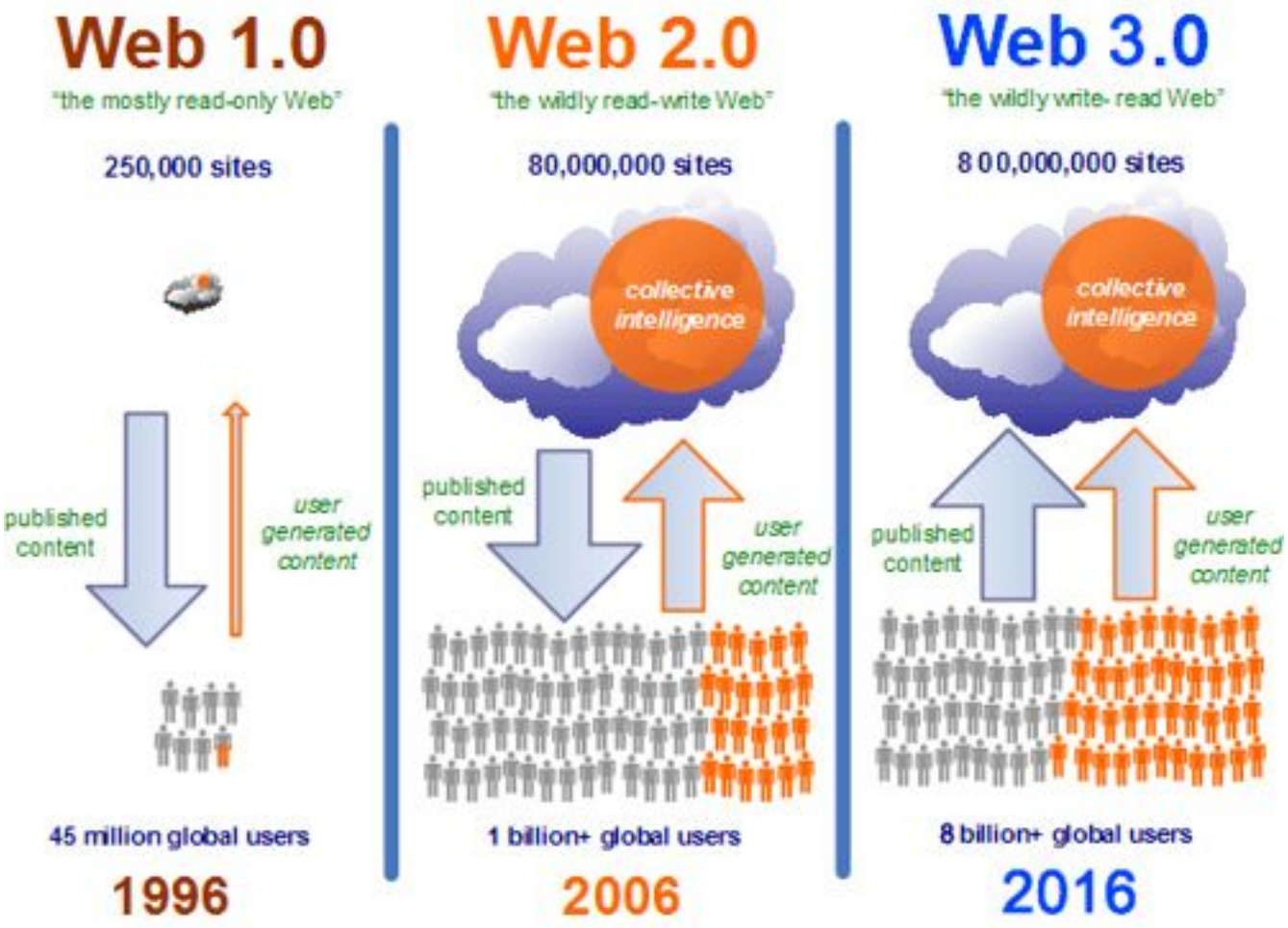
What is the difference between the Internet and the WWW?

Internet

- A network of thousands (or more) of computer networks
- Existed before the WWW
- The network infrastructure that makes the WWW work
- TCP/IP Protocol

The WWW

- The application layer built on top of the Internet
- A service that provides access to web pages
- Anything accessible via a URL

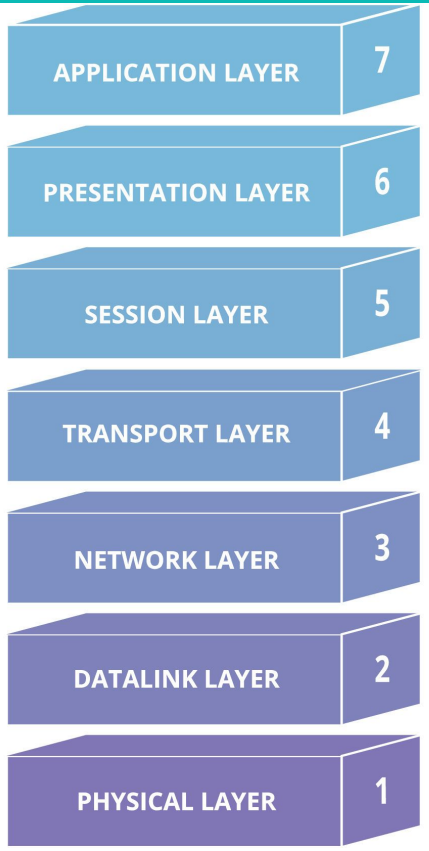

Web 1.0
"the mostly read-only Web"
250,000 sites
published content
user generated content
45 million global users
1996
Web 2.0
"the wildly read-write Web"
80,000,000 sites
collective intelligence
published content
user generated content
1 billion+ global users
2006
Web 3.0
"the wildly write- read Web"
8 0 0,000,000 sites
collective intelligence
published content
user generated content
8 billion+ global users
2016

HTTP

Hypertext Transfer Protocol (HTTP)

***HTTP** is a protocol which allows the fetching of **resources**, such as **HTML** documents. It is the foundation of any data exchange on the Web and a **client-server protocol**, which means requests are initiated by the recipient, usually the Web browser. A complete document is reconstructed from the different sub-documents fetched, for instance text, layout description, images, videos, scripts, and more. - HTTP Overview MDN web docs*

HTTP is an *application layer* protocol



— Human-computer interaction layer, where applications can access the network services

— Ensures that data is in a usable format and is where data encryption occurs

— Maintains connections and is responsible for controlling ports and sessions

— Transmits data using transmission protocols including TCP and UDP

— Decides which physical path the data will take

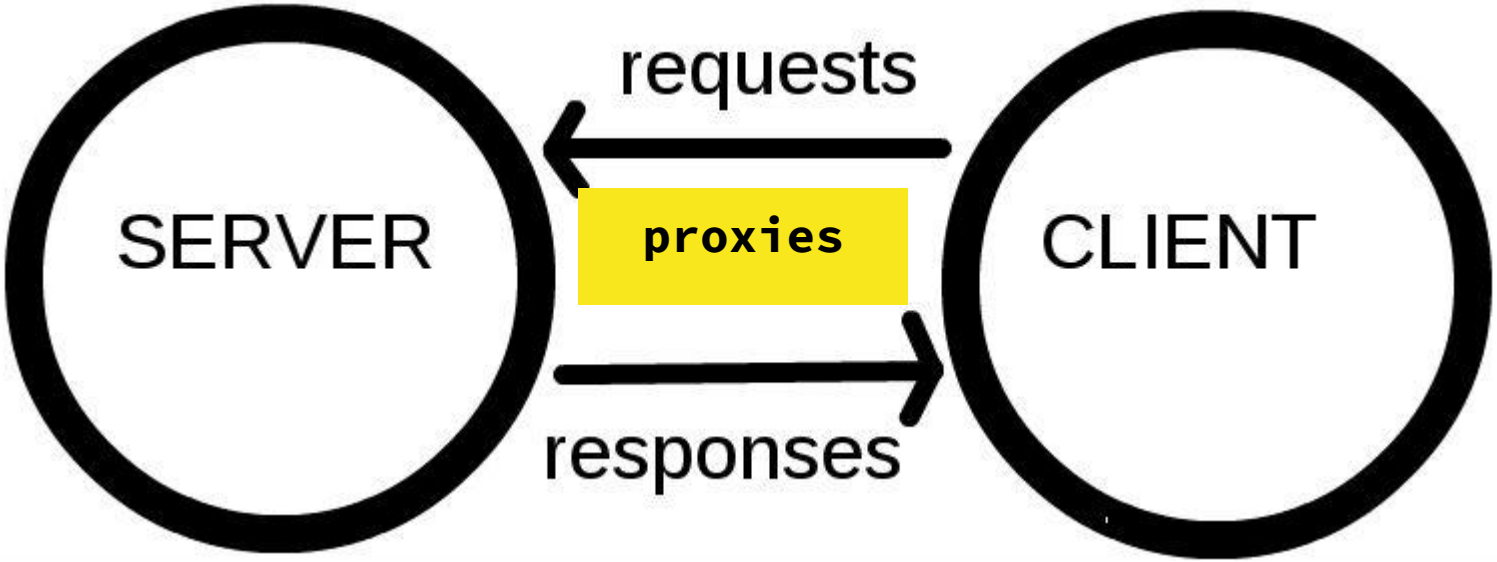
— Defines the format of data on the network

— Transmits raw bit stream over the physical medium

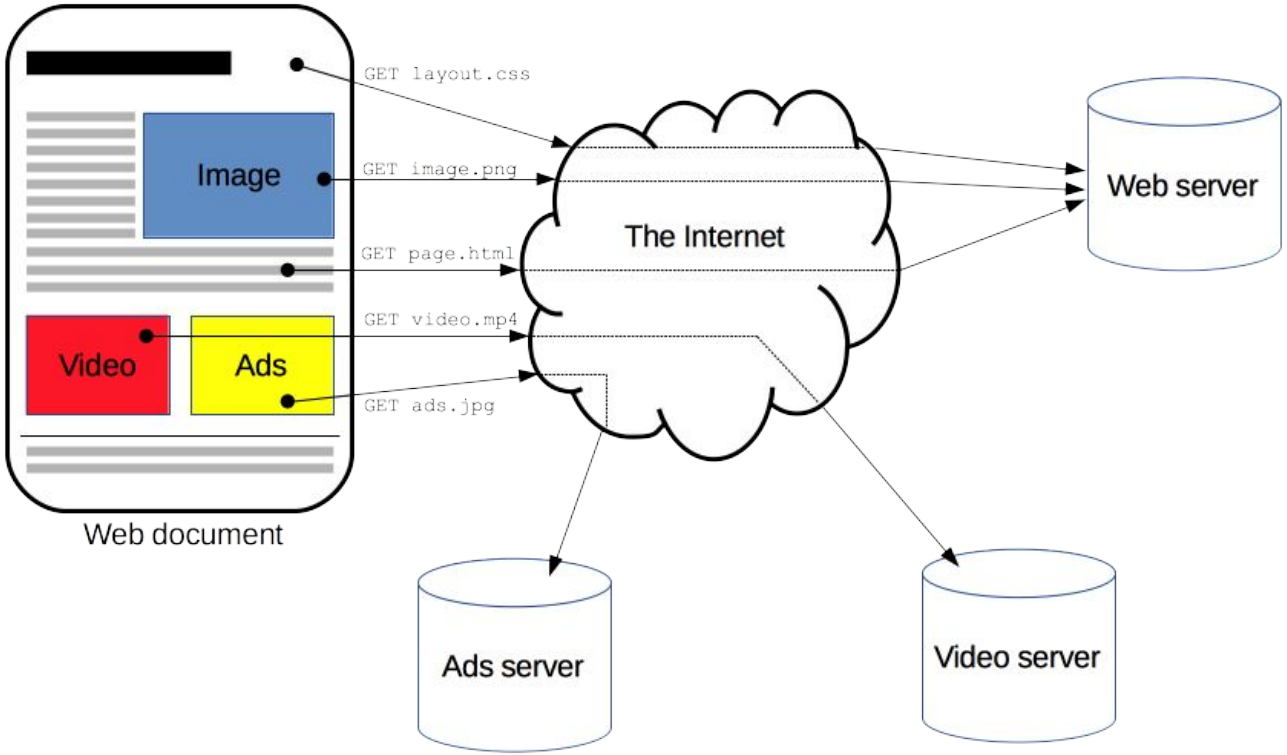
Features of HTTP

- HTTP is simple
  - The messages are designed to be human readable
- HTTP is extensible
  - HTTP/1.0 introduced headers which allow additional functionality
- HTTP is stateless
  - There is no link between two successive requests
- HTTP is stateful
  - HTTP Cookies allow HTTP requests to share the same context

Client Server Architecture

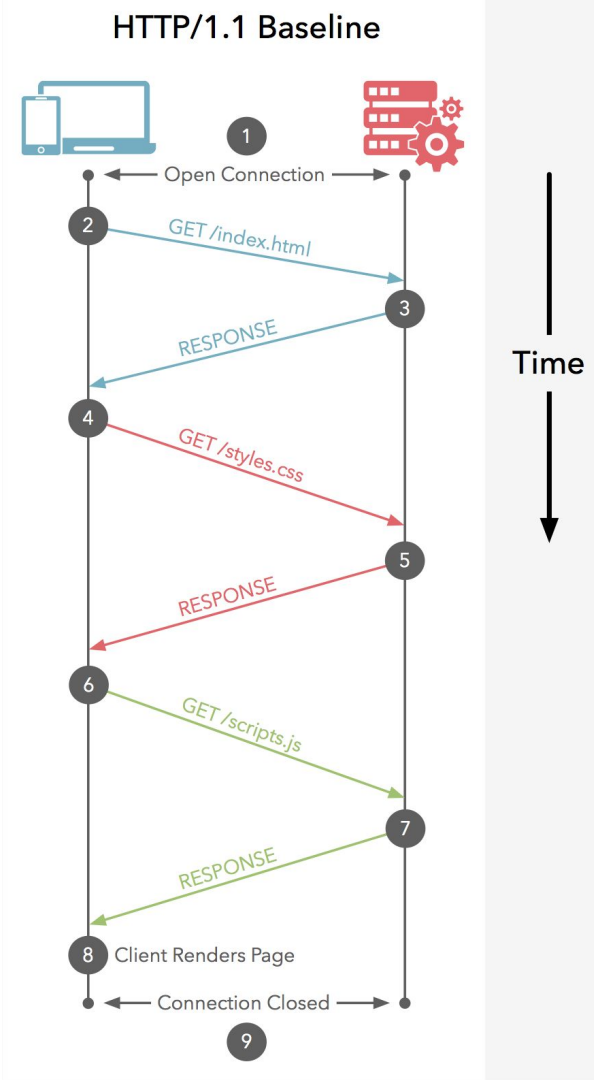


High Level Perspective



HTTP Flow

1. Open connection
2. Send HTTP request for resource
3. Get response from server
4. Send subsequent response for additional resource
5. Get response from server
6. Send subsequent response for additional resource
7. Get response from server
8. Render document
9. Close the connection



Elements of HTTP

- Request Methods - Verbs
  - GET - Requests a representation of a specific resource. Retrieve only.
  - POST - Submit an entity to a specified resource, often causing a change in state on the server.
  - PUT - Replace the current representation of the specified resource with the request payload.
  - DELETE - Remove the specified resource from the server.
  - HEAD - Same as GET, but without the response body.
  - *CONNECT, OPTIONS, TRACE, PATCH*
- User Agent - Information about the application making the request
- Headers - Metadata about the request
- Body - Data sent or received

Resources: URIs, URNs, URLs

- The target of HTTP requests is called a “resource”
- Resources can be documents, photos, or anything.
- Resources are identified by a Uniform Resource Identifier (URI)
- Uniform Resource Locators or URLs are the most common kind of URI

URL syntax

- A URL is composed of five parts
  - A protocol specification (http, ftp, gopher)
  - An authority (domain name) (google.com, pitt.edu)
  - A port specification (usually defaults to :80 for http and :443 for https)
  - A relative path specifier (/docs/web/http, /~edmonds, /)
  - A fragment identifier (#schedule) or query (?q="Cool search terms") Anchor or parameters
- Tim Berners-Lee says "Cool URIs don't change"


www.example.com
Domain Name
http://
Protocol
:80
Port
/path/to/myfile.html
Path to the file

?key1=value1&key2=value2

Parameters

#SomewhereInTheDocument

Anchor

HTTP Status Code

- HTTP has five categories of status code
  - 1xx: informational – used for development
  - 2xx: Successful response
  - 3xx: Redirection
  - 4xx: Client Error
  - 5xx: Server Error

- Frequently used codes:
  - 200 - success
  - 301 and 302 - Moved permanently or temporarily
  - 400 - bad request
  - 401 - unauthorized
  - 403 - forbidden
  - 404 - not found

Multipurpose Internet Mail Extension (MIME) Types

- The contents of resources are specified using
- Whether the browser can handle them is not the concern of the server
- Some common content-types are:

`text/plain`

`text/html`

`text/javascript`

`text/css`

`image/jpeg`

`image/png`

`audio/mpeg`

`audio/ogg`

`audio/*`

`video/mp4`

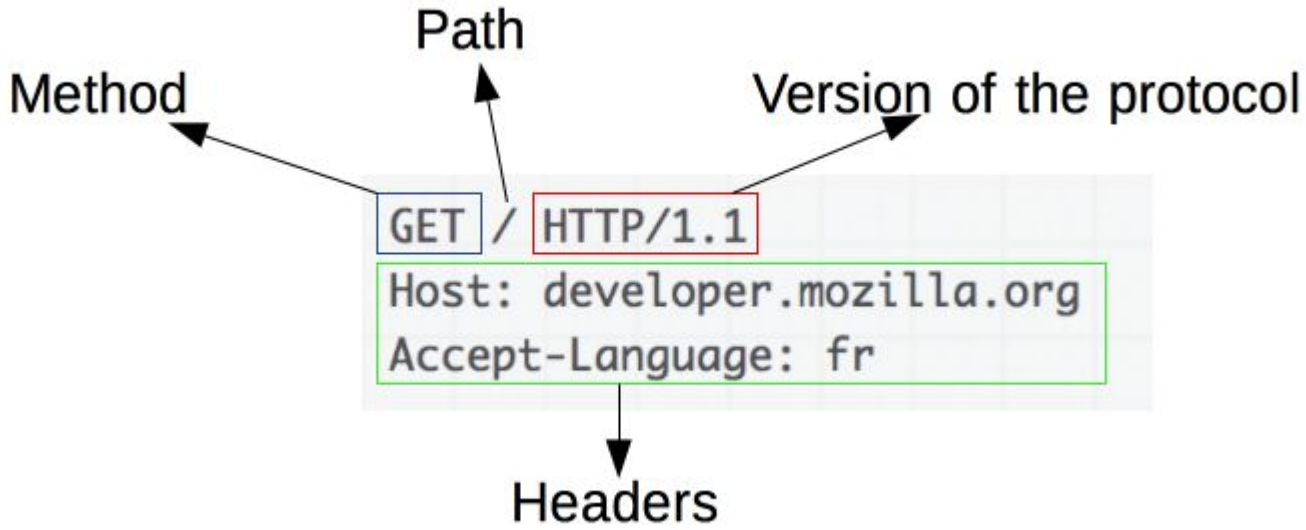
`application/*`

`application/json`

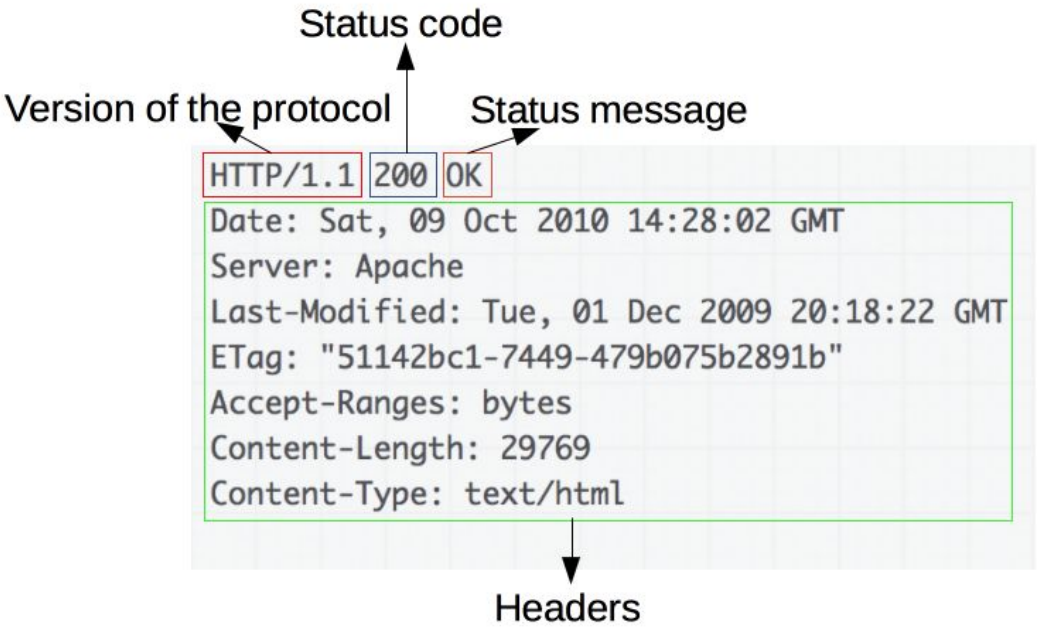
`application/ecmascript`

`application/octet-stream`

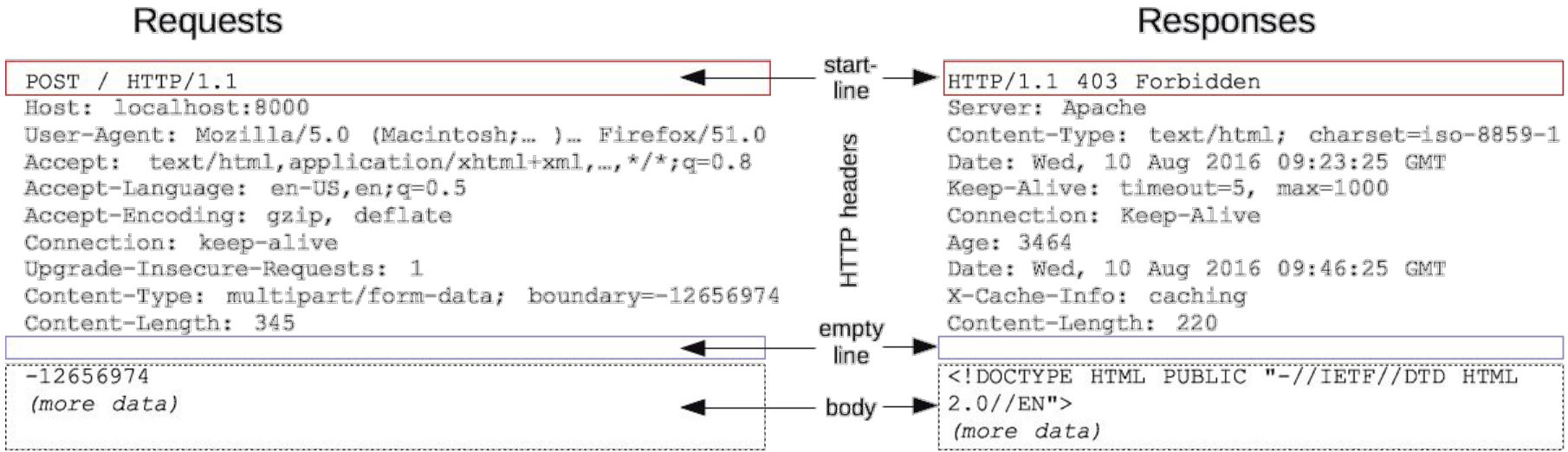
HTTP Requests



HTTP Response



HTTP Requests



Activity

What is Glitch?

- Glitch is a platform and community for creating web applications
  - No deployment headaches, focus on the code and the app, not the infrastructure
- It lets you create, edit, remix, and share full-stack web applications
- You are going to use Glitch this term for the activities and assignments
- https://glitch.com/about/features/

Understanding the Glitch Platform

- Every Glitch user has their own *user page*
  - ***https://glitch.com/@{username}***
- Every new or remixed Glitch project has a unique name, which you can change (if it is unique)
  - ***sun-accordion***
  - ***glorious-epoxy***
- Every Glitch project has multiple perspectives
  - Project home page: ***https://glitch.com/~{project-name}***
  - Live URL: ***https://{project-name}.glitch.me***
  - Editing interface: ***https://glitch.com/edit/#!/{project-name}***

Activity

1. Create an account on https://glitch.com
   a. Click “Sign In” in the upper right and select an option (Facebook, Github, Email)
2. Fill out your Glitch Profile and upload an Avatar image.
3. Spend some time browsing around the Glitch website. Look at the different categories of apps (Hello Worlds, Music, Games, Tools for Work)
4. **Think about** what type of apps you would like to make in this class. Find 2-3 related Glitch apps. Write a short paragraph about each project interests you.
5. Try **remixing** an existing app (click the “Remix This” button to create your own copy of the app) and editing the source code. See what happens!
6. Submit to CourseWeb in the “Activity 1 - Glitch”
   a. A URL pointing to your Glitch *user page*

NO CLASS next week.