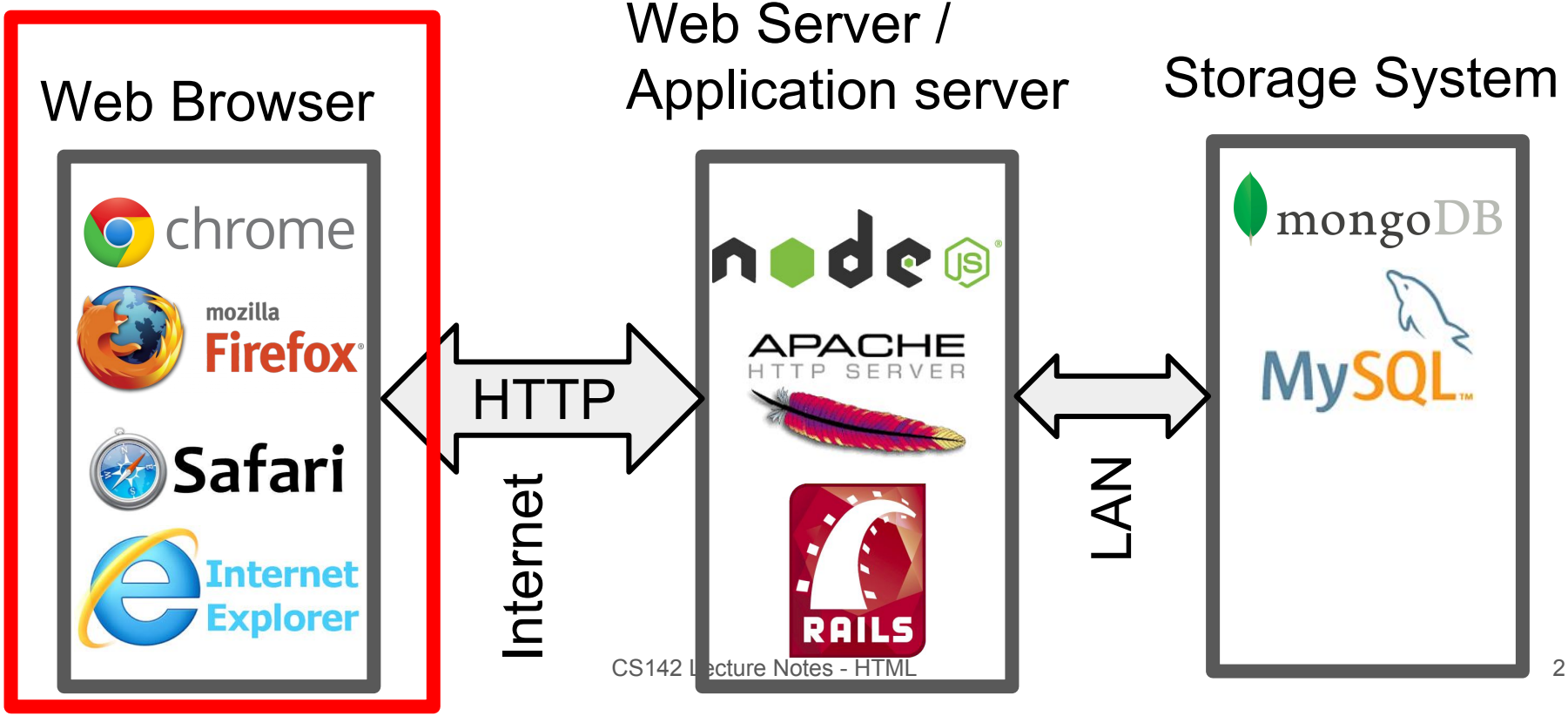
HyperText Markup Language (HTML)

Mendel Rosenblum

Web Application Architecture



Browser environment is different

Traditional app: GUIs based on pixels

- Since 1970s: software accessed mapped framebuffers (R/G/B)
- Toolkits build higher level GUI widgets (buttons, tables, etc.)

Web browsers display **Documents** described in **HTML**

- Until the most recent HTML5's canvas region, you couldn't write pixels
- Make applications out of documents
  - Early web apps: Multiple documents (pages) with 'form' tag for input
  - Current: Use JavaScript to dynamically generate and update documents

HTML: HyperText **Markup Language**

Concept: **Markup Language** - Include directives with content

Directives can dictate presentation or describe content

Idea from the 1960s: RUNOFF

Examples: italics word, <title>Title words</title>

Approach

1. Start with content to be displayed
2. Annotate it with **tags**

HTML uses < > to denote tags

HTML tags

Tags can provide:

Formatting information (`<i>` for italic)

Meaning of text:

- `<h1>` means top-level heading
- `<p>` means paragraph
- `<ul><li>` for unordered (bulleted) list

Additional information to display (e.g., `<img>`)

Tags can have tags inside (nesting supported)

Example of HTML - Start with raw content text

```
Introduction
There are several good reasons for taking
CS142: Web Applications:
You will learn a variety of interesting concepts.
It may inspire you to change the way software is developed.
It will give you the tools to become fabulously wealthy.
```

Example of HTML - Annotate with tags

```
<h2>Introduction</h2>
<p>
There are several good reasons for taking
<i>CS142: Web Applications</i>:
</p>
<ul>
<li>
You will learn a variety of interesting concepts.
</li>
<li>
It may inspire you to change the way software is developed.
</li>
<li>
It will give you the tools to become fabulously wealthy.
</li>
</ul>
```

Browser doesn’t care but programmers do

```
<h2>Introduction</h2>
<p>
  There are several good reasons for taking
  <i>CS142: Web Applications</i>:
</p>
<ul>
  <li>
    You will learn a variety of interesting concepts.
  </li>
  <li>
    It may inspire you to change the way software is developed.
  </li>
  <li>
    It will give you the tools to become fabulously wealthy.
  </li>
</ul>
```

Example HTML - Browser output

Introduction

There are several good reasons for taking *CS142: Web Applications*:

- You will learn a variety of interesting concepts.
- It may inspire you to change the way software is developed.
- It will give you the tools to become fabulously wealthy.

HTML Evolution

Influenced by browser implementation quirks

What to do if you see “ Some text” (missing closing)?

1. Complain bitterly about malformed HTML.
2. Figure out there was a missing , add it, and continue processing.

Forked into HTML and XHTML (XML-based HTML)

XHTML is more strict about adhering to proper syntax

For the HTML class projects (1, 2, and 3) we will use XHTML

Users came to depend on browser quirks, so browsers couldn't change

Example XHTML document

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
  <head>
    <title>Hello World</title>
  </head>
  <body>
    <p>Hello world!</p>
  </body>
</html>
```

Basic Syntax rules for XHTML

Document: hierarchical collection of **elements**, starting with `<html>`

- Element: start tag, contents, end tag
- Elements may be nested
- Every element must have an explicit start and end
  - Can use <foo /> as shorthand for <foo></foo>
- Start tags can contain **attributes**:

```
<img src="face.jpg">
<input type="text" value="94301" name="zip">
<div class="header">
```

Need to handle markup characters in content

To display a literal < or > in a document, use entities:

`<`	Displays <
`>`	Displays >
`&`	Displays &
`&quot;`	Displays "
` `	Nonbreaking space (won't insert a line break at this space)

Many other entities are defined for special characters.

Whitespace is not significant except in a few cases (e.g. `textarea`, `pre` tags)

Example XHTML document structure

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
  <head>
    <title>Hello World</title>
  </head>
  <body>
    <p>Hello world!</p>
  </body>
</html>
```

XHTML document structure

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
```

Indicate that this is an XHTML document, conforming to version 1.0 of the standard; use these lines verbatim in all the web pages you create for this class.

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
```

Outermost element containing the document

<head>: Contains miscellaneous things such as page title, CSS stylesheets, etc.

<body>: the main body of the document

Common usage XHTML tags

`<p>`	New paragraph
` `	Force a line break within the same paragraph
`<h1>, <h2>, ...`	Headings
`<b>, <i>`	Boldface and italic
`<pre>`	Typically used for code: indented with a fixed-width font, spaces are significant (e.g., newlines are preserved)
`<img>`	Images
`<a href="...">`	Hyperlink to another Web page

Common used XHTML tags - continued

<table>, <tr>, <td>	Tables
`<ul>, <li>`	Unordered list (with bullets)
`<ol>, <li>`	Ordered list (numbered)
`<div>`	Used for grouping related elements, where the group occupies entire lines (forces a line break before and after)
`<span>`	Used for grouping related elements, where the group is within a single line (no forced line breaks)
`<form>, <input>, <textarea>, <select>, ...`	Used to create forms where users can input data

Commonly used tags: <head> section

<title>	Specify a title for the page, which will appear in the title bar for the browser window.
<link>	Include CSS stylesheets
<script>	Used to add Javascript to a page (can be used in body as well)

HTML differences from XHTML

HTML supports the same tags, same features, but allows quirkier syntax:

- Can skip some end tags, such as `</br>`, `</p>`
- Not all attributes have to have values: `<select multiple>`
- Elements can overlap: `<p><b>first</p><p>second</b> third</p>`

Early browsers tried to "do the right thing" even in the face of incorrect HTML:

- Ignore unknown tags
- Carry on even with obvious syntax errors such as missing `<body>` or `</html>`
- Infer the position of missing close tags
- Guess that some < characters are literal, as in "What if x < 0?"
- Not obvious how to interpret some documents (and browsers differed)

Example HTML of an simple Angular Web App

```
<!doctype html>
<html ng-app="cs142App" ng-controller="MainController">
    <head>
        <title>CS142 Class Project</title>
        <script src="node_modules/angular/angular.js"></script>
        <script src="mainController.js"></script>
        <link rel="stylesheet" type="text/css" href="main.css" />
    </head>
    <body>
       <h1>CS142 Project #2 - JavaScript Calisthenics</h1>
       <p>You should create three local files for your problem solutions:</p>
        <ul>
            <li>Problem #1 - Getting started -
                <a href="getting-started.html">getting-started.html</a>
            </li>
       </ul>
    </body>
</html>
```