CSS3
Cascading Style Sheets

Browser Default Styling

- The same html file may look different when viewed on different browsers.
  - Some tags are supported, some aren’t
  - Browsers may have different *default styles*
- In general, default looks are plain.

Adding Style

- As styling tags were phased out of html, styling was done with style attribute

```
<h1 style = "color:blue">Styled Heading</h1>
```

Styled Heading

- Violated separation of content/style

Cascading Style Sheet

- CSS defined generic rules that can apply to multiple elements

```
selector {
    property: value;
}
```

```
h1 {
    color: blue;
}
```

Styled Heading

Rule Syntax

- Brackets and semicolons are very important
- This is where a good editor can make a BIG difference

```
h1 {
    color: blue;
    /* This is a CSS comment */
}
```

Multiple Properties

```
h1 {
    color: blue;
    background-color: yellow;
}
```

Internal Style Sheet

```
<head>
    <title>Home</title>
    <style>
        h1 { color: blue;}
    </style>
</head>
```

- Styling is defined within <head>
- Rules are defined within <style>
- Styles are applied to all elements in that file

Don't forget to close the style tag!!

External Style Sheet

- Rules are in an external file (no style tag)
- A link to the style sheet is put in the head section of the html file.

  <link rel="stylesheet" href="css/style.css">

- Styles are applied to all elements in all files that link to the style sheet

IMPORTANT

- You should primarily use an external style sheet in this class – rare exceptions may occur
- You should not create a separate style sheet for each page. Typically, a rule should only appear once.

Color Conventions

- Color names
  blue, red, yellow, etc.

blue

- Hexadecimal
  #0000FF, #FF0000, #FFFF00

#0000FF #FF0000 #FFFF00

- rgb
  (0, 0, 1), (1, 0, 0), (1, 1, 0)

rgb(0,0,1)

- rgba
  (0, 0, 1, .5)

rgba(0, 0, 1, .5)

Accessibility

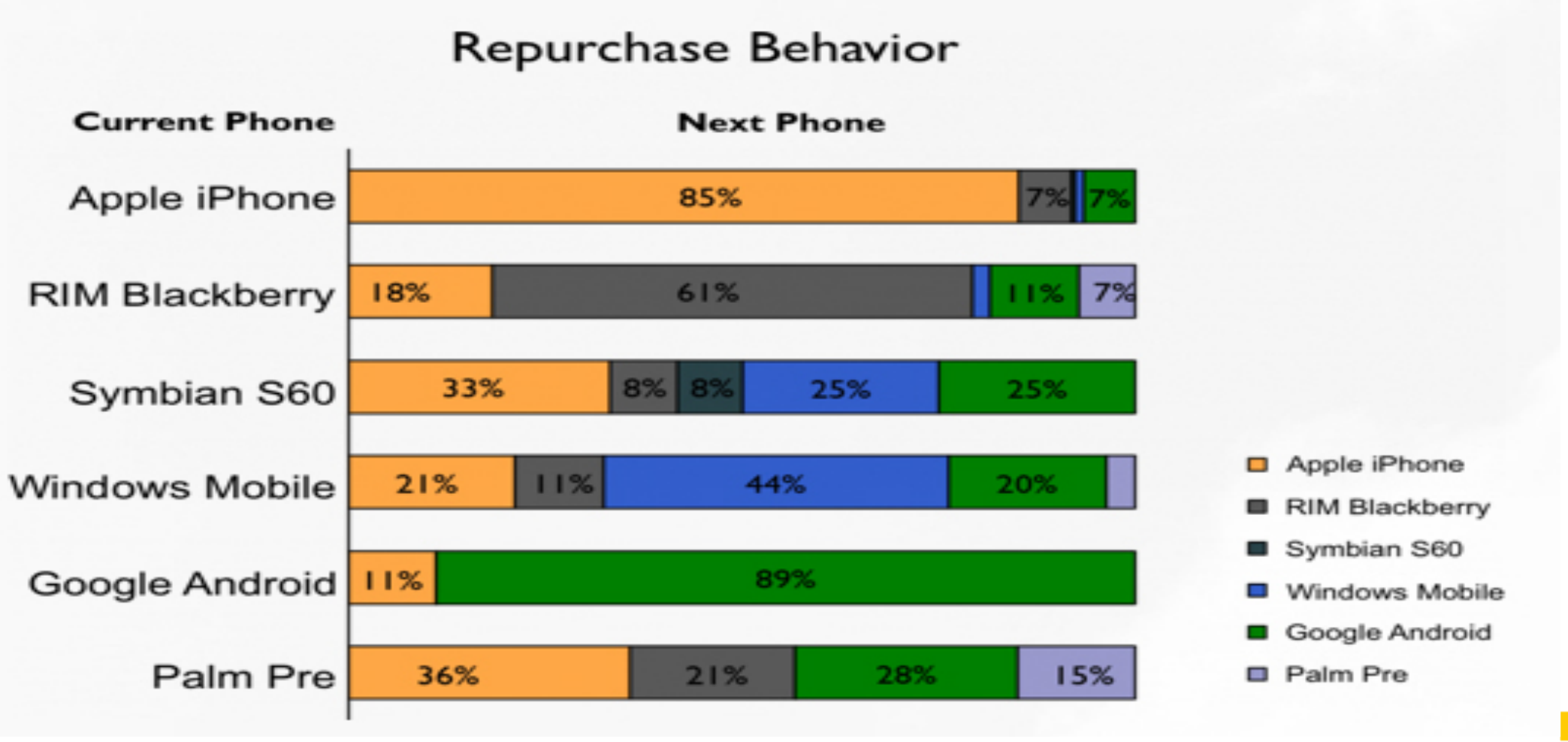
- Appropriate use of color is critical to web accessibility

- Many more people are visually impaired or color blind than are legally blind

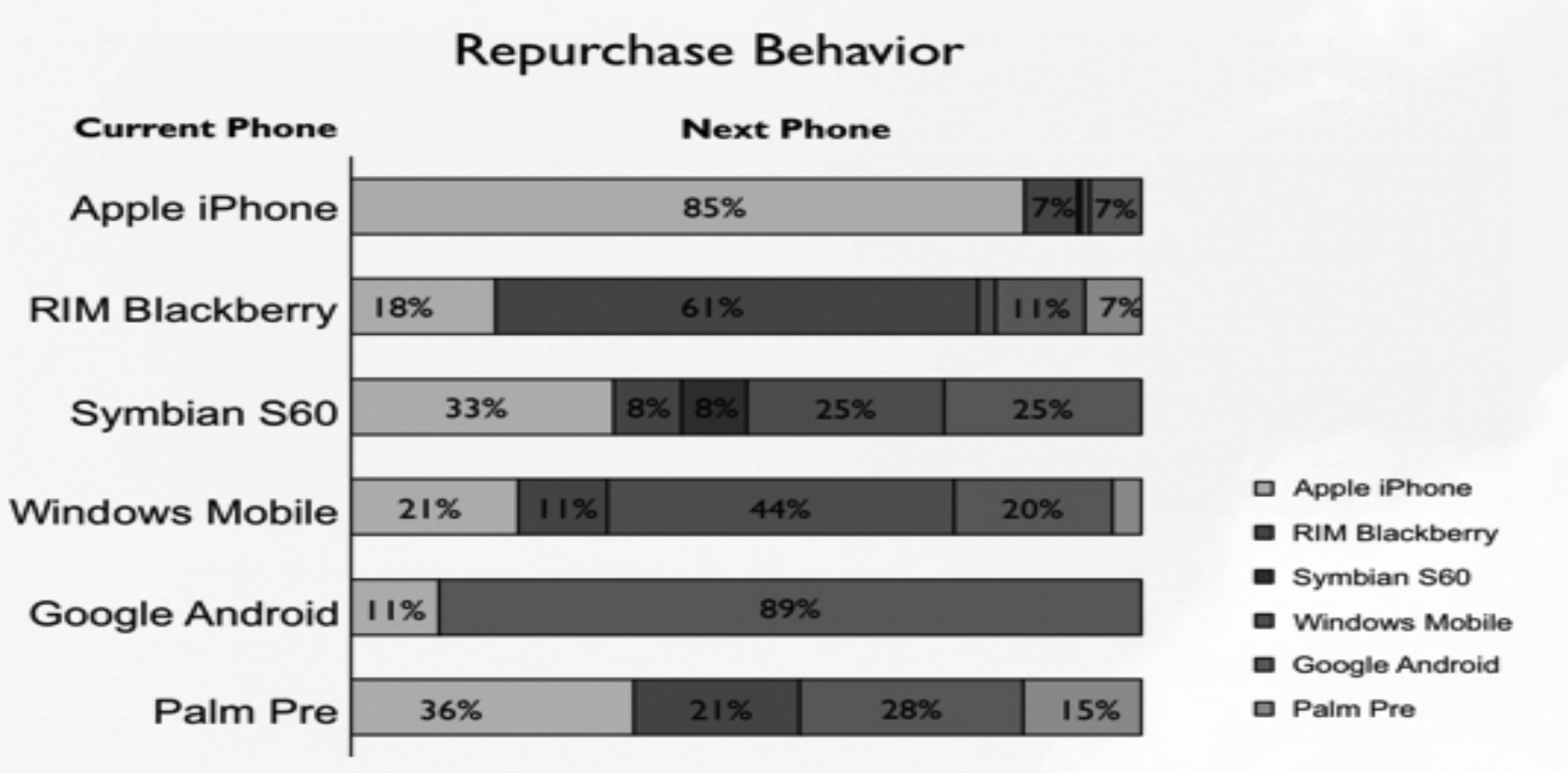
What is color contrast?

- Your judgement is not sufficient so use validators that quantify the contrast between text and its background
- WCAG AA requires "at least 4.5:1" contrast
- If you ***LOVE*** your color scheme, make the font bigger.

Don’t use color alone to convey meaning



Test in gray scale …



Styling Your Text

- font (family, style, variant, size)
- color and background
- alignment
- line-height
- more...

font-family

- Font families are styles of text
- Examples:
  - Helvetica, Courier, "Courier New", "Comic Sans MS", *cursive*, Verdana

Defining Multiple font-family values

- Not all font-families are supported by all of the operating systems, so you can provide alternatives.

```
h1 {
    font-family: Courier, Impact, Arial;
}
```

Google Fonts

```
<link href='https://fonts.googleapis.com/css?family=Anton'
                                        rel='stylesheet'>
<style>
    body {
        font-family: 'Anton';
    }
<style>
```

- https://www.w3schools.com/howto/howto_google_fonts.asp

The “Cascading” part of CSS

- Browser default
- *External* (in a separate file)
- *Internal* style (in the head section)
- *Inline* style (inside an HTML element)

Rule precedence

```
h1 {  color: blue;
      font-family: Arial !important;}
.....
h1 {  font-family: Times; }
```

- What if a selector appears more than once in a file?
- What if one selector is defined in two external files?
- It is possible to override default precedence.

Review

- Why do we want/need to separate content from formatting?
- How does this also tie in to external/internal style sheets?
- Understand that this is very powerful. See http://www.csszengarden.com/

Designing for Accessibility

- POUR Principle
  - Perceivable
  - Operable
  - Understandable
  - Robust

Perceivable

- Provide text alternatives for images
- Provide captions and transcripts for video and audio
- Use correct semantic markup so content can be presented in different ways

  Make it easier for users to see content by using good color contrast

Operable

- ***All functionality available from the keyboard!***
- Users have control over timing and limits
- Do not cause seizures (don’t flash content)
- Provide ways to help users navigate, find content, and determine where they are

Input Devices

Understandable

- Economical and plain use of language
- Text supplemented with illustrations, videos, and other formats where appropriate (i.e., use good Universal Design)
- Navigation, information structure are discernable and consistent
- Make pages operate in predictable ways
- Help users avoid and correct mistakes

Robust

- Is your site functional across various technologies (smart phone, screen reader, laptop, pensticks, etc..)?
- Syntax errors that don’t affect visual presentation may hamper assistive technology and accessibility tools
- Adhering to W3C standards ensures future compatibility
- Validate your code!!!

Accessibility Review

- Accessibility starts with proper HTML tags
- Styling can actually make it HARDER for some people to access the information
- Get into the early habit of utilizing accessibility tools
- “Cool” new style should not be at the cost of accessibility

Advanced Selectors

- We have focused on *type* selectors but it is possible to style more specific elements

CSS DOM Selectors

- Descendant selector

```
nav li {
    list-style-type: none;
}
```

- Child selector

```
nav > a {
    list-style-type: none;
}
```

id Selectors

```
#mainLogo {
    border: 5px solid #0006CC; }
```

- Identifies a single element in the DOM.
- Historically used for , , etc.
- There is a small movement to move the use of id ***out*** of CSS

class Selector

```
.thumb {
    border: 1px solid #0006CC;
    width: 200px;}
```

- Identifies element(s) in the DOM that are part of a common group.
- Think of thumbnail images, all of the links that are in the navigation, your social media images, etc….

classes vs. ids

- Syntax is “.” and “#”
- classes can be used multiple times
- id should be unique
- ids take precedence over classes in styling (see more later)

Changing the Scope

- p.main → paragraphs using main class
- header img.thumbail → paragraphs inside header that have class="thumbnail" attribute.
- p, h1, #history, .special, → rules to apply to all of them

More Attribute Selectors

- Universal
  - * applies styling to every element on the page
  - Ackk!! Try this!
- Attribute Selectors
  - a[href=‘info.html’]
- PseudoClasses
- Pseudo Elements

Attribute selectors

- You can search for elements that have specials attributes by using operators:

  ^ : match the beginning exactly
  a [href^='http://umich']

  $: match the end exactly
  img[src$ = '.png'] → apply to .png images

  * : wildcard
  a [href*='umich']

Pseudo-Classes

- Elements that are dynamically populated or dependent on tree structure
  - focus, hover, first-of-type, last-child

```
.a:hover {
    border: 2px solid #0006CC;
}
```

Types of Pseudo-Classes

- Structural:
  :nth-child, empty, only-of-type, last-of-type
- Link
  :link, :visited
- User Action
  :hover, :active, :focus
- Forms (interfaces)
  :enabled, :checked, :disabled

https://www.w3schools.com/css/css_pseudo_classes.asp

Pseudo-Elements

- These elements aren’t part of the DOM
- Can be used to style specific (unique) parts of the page

Types of Pseudo-Elements

- Textual
  :first-letter, :first-line
- Positional/Generated**
  :before :after
- Fragments
  ::selection

**Reference this slide for the JS video homework.

States

- Some links are blue, some are purple, etc. Why???
  - a:link: a normal, unvisited link
  - a:visited: has been visited
  - a:hover: activated by mouse (touchscreens….?)
  - a:focus: activated with the keyboard
  - a:active: is being clicked

Example

- Examples:
  - AdvancedSelectors.html
  - AdvancedSelectors2.html

Rule Specificity

- Each rule has a weight made up of four digits.
  1. +1 for or each element and pseudo-element
  2. +10 for each attribute, class or pseudo-class
  3. +100 for each id
  4. +1000 if the style attribute is used

Specificity example

A. li { }
B. <li style="color:#FFFF00">Pizza
C. nav ul.intro li{}
D. #content li { }

- https://www.w3schools.com/css/css_specificity.asp

Cascading Review

- One element can have many classes and ids associated with it

<li class = “special early dark” id = ”main”/>

- Browser “starts at the top” and applies each rule, sometimes overriding earlier rules.

The Good News

- You can use style sheets from others to style your code, just by adding class!!
- You can override style sheets from others just by rewriting the class, or making your own version of it and linking it last.

Example: Anchor Links

- Links can take on all of the usual styles as well as *text-decoration*

This is a link

```
a {
    display: block;
    font-weight: bold;
    color: #ffffff;
    background-color: #0006CC;
    width: 200px;
    text-align: center;
    padding: 4px;
    text-decoration: none;
}
```

This is a link

Example: Lists

- Number of properties beyond font, margin, etc.
  - list-style-type
  - list-style-image
  - list-style-position
  - list-style

List and Link Examples

- Examples:
  - lists.html
  - links.html

“Buttons”

- Many designers try to make their links look like buttons – DON’T!
- Be semantic, if you want a button use the <button> element instead.

```
<button>Click Me!</button>
```

Acknowledgements

- Initial Development: Colleen van Lent , University of Michigan School of Information