THE MARKUP STRIKES BACK

The importance of good markup

Definitions

- “Markup” refers to HTML
- “Styling” refers to CSS

Getting it right

Why good markup is important, and how to write it.

Getting markup right

- Why is markup—getting markup right—important?

Getting markup right

- Why is markup—getting markup right—important?
  - It can be the hardest part of your application. No kidding.

Getting markup right

- Why is markup—getting markup right—important?
  - It can be the hardest part of your application. No kidding.
  - **Readability**

Getting markup right

- Why is markup—getting markup right—important?
  - It can be the hardest part of your application. No kidding.
  - Readability
  - **Maintainability**

Getting markup right

- Why is markup—getting markup right—important?
  - It can be the hardest part of your application. No kidding.
  - Readability
  - Maintainability
  - **Predictability**

Getting markup right

- Why is markup—getting markup right—important?
  - It can be the hardest part of your application. No kidding.
  - Readability
  - Maintainability
  - Predictability
  - **Compatibility**

Writing good markup

- Separate content from presentation
  - ```

      <font color="red" size="3”>
        Danger!
      </font>

    ```
    ```

Writing good markup

- Separate content from presentation
  - ~~ Danger! ~~
  - `<p class="alert">Danger!</p>`

```
p.alert
{
  color: red;
  font-size: 140%;
  text-align: center;
}
```

Writing good markup

- Separate content from presentation
  - ```
    <a href="#" onclick="showHide();">
      Show/hide stuff.
    </a>

      Stuff.

    function showHide()
    {
        toggle($('toggle-me'));†
    }
    ```
    ```

Writing good markup

- Separate content from presentation
- **Follow the standards**

Writing good markup

- Separate content from presentation
- Follow the standards
- **Organize your code**

Writing good markup

- Separate content from presentation
- Follow the standards
- Organize your code
- **Don’t repeat yourself**

Writing good markup

- Separate content from presentation
- Follow the standards
- Organize your code
- **Don't repeat yourself**

```
<p style="background-color: #cccccc;">
    Some content.
</p>
<p style="background-color: #cccccc;">
    Some more content.
</p>
<p style="background-color: #cccccc;">
    Yet more content.
</p>
```

Writing good markup

- Separate content from presentation
- Follow the standards
- Organize your code
- Don’t repeat yourself
- **Don’t take shortcuts**

Writing good markup

- Separate content from presentation
- Follow the standards
- Organize your code
- Don't repeat yourself
- Don't take shortcuts
- **Comment your code!**

Writing good markup

- Separate content from presentation
- Follow the standards
- Organize your code
- Don't repeat yourself
- Don't take shortcuts
- **Comment your code!**

```
<div class="blah">
…
</div> <!-- /blah -->
```

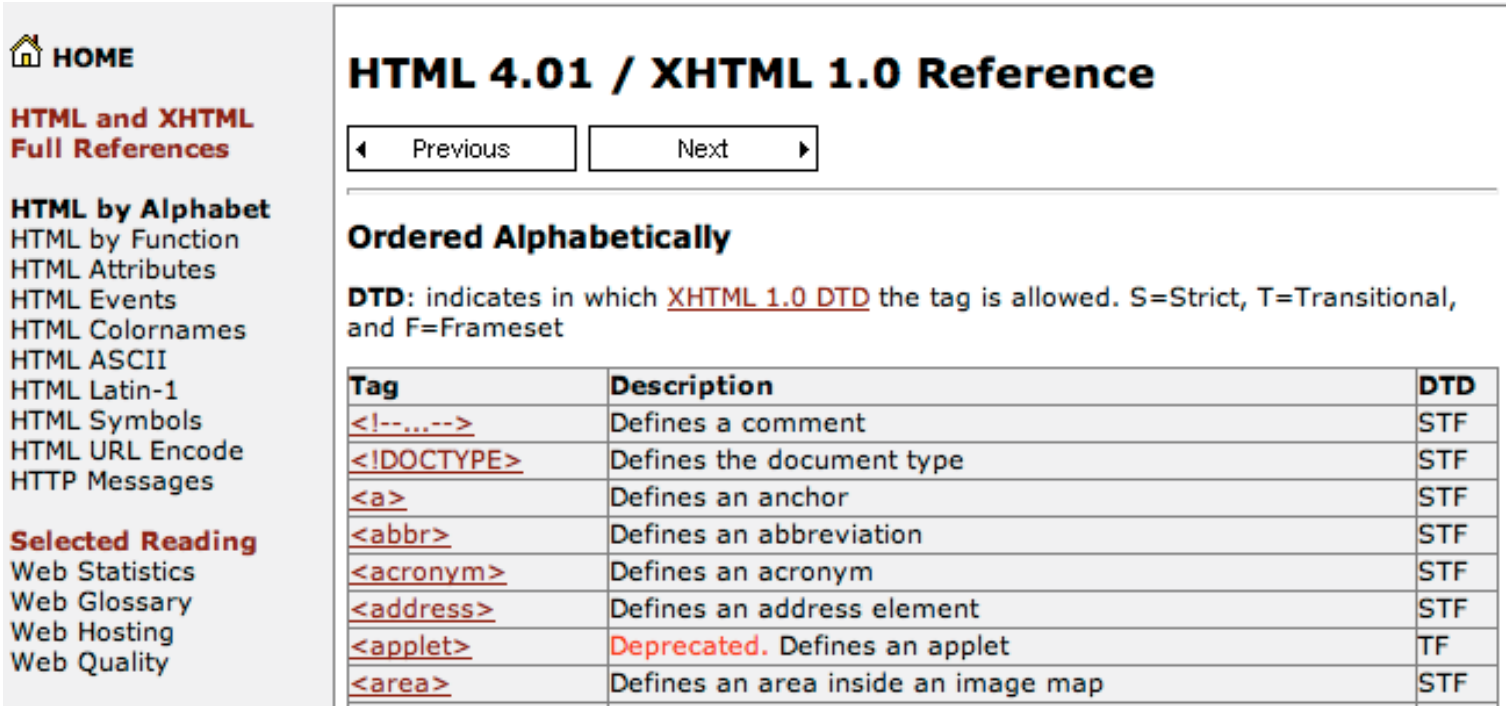
The standards

The standards

- Official definitions at W3C*: HTML 4.01, XHTML 1.0, CSS 2.1
  - Good for reference, not for learning
- Learn by doing tutorials at sites like
  - W3Schools: HTML, XHTML, CSS, JavaScript, HTML DOM
  - Or WestCiv: CSS

The standards (cont'd)

- W3Schools also provides convenient reference guides



HOME

HTML and XHTML Full References

HTML by Alphabet
HTML by Function
HTML Attributes
HTML Events
HTML Colornames
HTML ASCII
HTML Latin-1
HTML Symbols
HTML URL Encode
HTTP Messages

Selected Reading
Web Statistics
Web Glossary
Web Hosting
Web Quality

HTML 4.01 / XHTML 1.0 Reference

Previous Next

Ordered Alphabetically

DTD: indicates in which XHTML 1.0 DTD the tag is allowed. S=Strict, T=Transitional, and F=Frameset

Tag	Description	DTD
<!--...-->	Defines a comment	STF
<!DOCTYPE>	Defines the document type	STF
<a>	Defines an anchor	STF
<abbr>	Defines an abbreviation	STF
<acronym>	Defines an acronym	STF
<address>	Defines an address element	STF
<applet>	Deprecated. Defines an applet	TF
<area>	Defines an area inside an image map	STF

Validate your markup

- How do you know your markup is right?
- Validate it! W3C markup and CSS validators ensure your code conforms to the specifications.

HTML & doctype

HTML vs. XHTML

- HTML 4.01 and XHTML 1.0 are functionally equivalent.
- XHTML follows XML syntax rules and is easier for developers to read.

Doctype

- What is doctype, and why is it important?

Doctype

- What is doctype, and why is it important?
  - Doctype is short for “document type declaration”, and it tells the validator/browser what flavor of markup you’re using[1].

Doctype

- What is doctype, and why is it important?
  - Doctype is short for “document type declaration”, and it tells the validator/browser what flavor of markup you’re using[1].
  - The doctype must appear at the top of the page

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN”
      "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<html>
<head>
  ...
</head>
<body>
  ...
```

Doctype

- What is doctype, and why is it important?
  - Doctype is short for “document type declaration”, and it tells the validator/browser what flavor of markup you’re using[1].
  - The doctype must appear at the top of the page
  - Bottom line: if you don’t have a doctype, you can’t validate your code, and you can’t guarantee that it will be rendered correctly. **Don’t forget the doctype.**

Doctype (cont'd)

- There are multiple doctypes, each representing a different flavor of markup: HTML 4.01 Strict, XHTML 1.0 Transitional, etc.
  - `<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">`[2]
  - `<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">`

Doctype (cont'd)

- There are multiple doctypes, each representing a different flavor of markup: HTML 4.01 Strict, XHTML 1.0 Transitional, etc.
  - `<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">`[2]
  - `<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">`
- Transitional doctypes are for those sites making the shift from older to modern markup[3], and strict doctypes are for markup that strictly adheres to the standards.

Doctype (cont'd)

- There are multiple doctypes, each representing a different flavor of markup: HTML 4.01 Strict, XHTML 1.0 Transitional, etc.
  - `<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">`[2]
  - `<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">`
- Transitional doctypes are for those sites making the shift from older to modern markup[3], and strict doctypes are for markup that strictly adheres to the standards.
- The goal: HTML 4.01 Strict/XHTML 1.0 Strict

Content type and encoding

- Doctype isn't enough; you need to specify the content (MIME) type and the file encoding as well.

Content type and encoding

- Doctype isn't enough; you need to specify the content (MIME) type and the file encoding as well.

Content type and encoding

- Doctype isn't enough; you need to specify the content (MIME) type and the file encoding as well.
- They are specified in the meta content type declaration in the header:
  - `<meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>`
  - Use `text/html` and `utf-8`[4,5].

Mini Q&A

CSS coding guidelines

CSS coding standards and organization

- Why is it important? Predictability and ease of troubleshooting.
  - ~6K lines of CSS in PSA alone

CSS coding standards and organization

- Why is it important? Predictability and ease of troubleshooting.
  - ~6K lines of CSS in PSA alone
- Split your CSS out into multiple files systematically.
  - E.g. core, layout, content, forms, page/function-specific, browser compatibility hacks

CSS coding standards and organization

- Why is it important? Predictability and ease of troubleshooting.
  - ~6K lines of CSS in PSA alone
- Split your CSS out into multiple files systematically.
  - E.g. core, layout, content, forms, page/function-specific, browser compatibility hacks
- Code to the standards, then hack for compatibility.

CSS selectors

- In CSS, **selectors** are pattern matching rules that determine which style rules apply to elements in the document tree[7].
  - `p.question` applies to `<p class="question">…</p>`
  - `div#address-form` applies to `<div id="address-form">…</div>`

CSS cascading

- The C in CSS stands for **cascading.**

CSS cascading

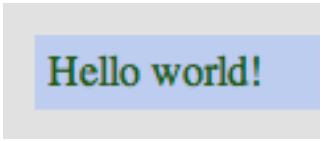
- The C in CSS stands for **cascading.**

```
22  <style type="text/css">¬
23    body¬
24    {¬
25      margin: 0px;¬
26      padding: 10px;¬
27      background-color: #e0e0e0; /*light grey*/¬
28      color: #606060; /*dark grey*/¬
29    }¬
30    ¬
31    p¬
32    {¬
33      margin: 2px;¬
34      padding: 5px;¬
35      background-color: #c0f0c0; /*light green*/¬
36      color: #006600; /*dark green*/¬
37    }¬
38    ¬
39    .greeting¬
40    {¬
41      background-color: #c0c0f0; /*light blue*/¬
42    }    ¬
43  </style>¬
```

```
50  <p class="greeting">¬
51    Hello world!¬
52  </p>¬
```

CSS cascading (cont'd)

- What is rendered:

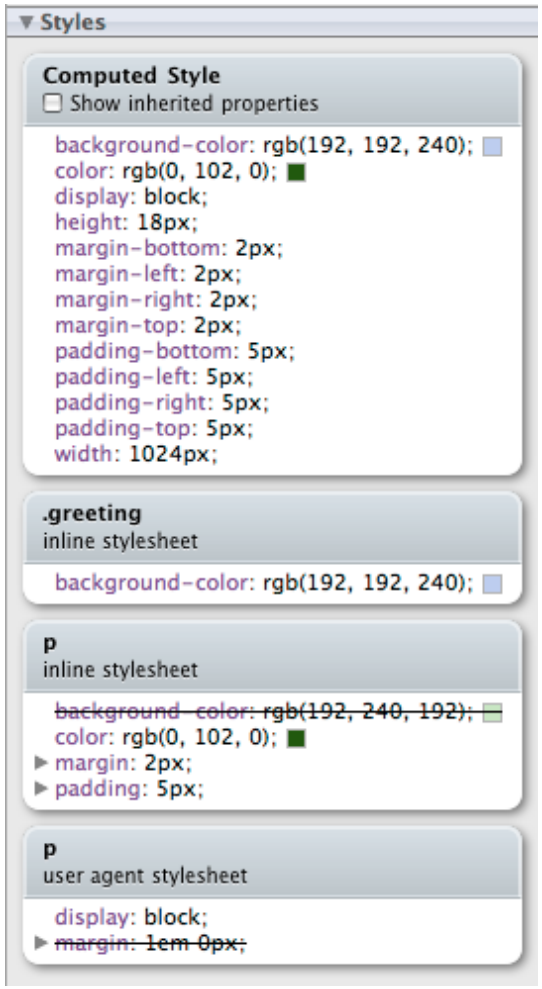


CSS cascading (cont'd)

- What is rendered:

Hello world!

- The cascade ➔



CSS specificity

- Style rules are ranked by **specificity.**

```
11  <style type="text/css">
12    body
13    {
14      margin: 0px;
15      padding: 10px;
16      background-color: #e0e0e0; /*light grey*/
17      color: #606060; /*dark grey*/
18    }
19
20    p
21    {
22      margin: 0px;
23      padding: 5px;
24      background-color: #c0f0c0; /*light green*/
25      color: #006600; /*dark green*/
26    }
27
28    .greeting
29    {
30      background-color: #c0c0f0; /*light blue*/
31    }
32
33    #intro p
34    {
35      background-color: #f0c0c0; /*pink*/
36    }
37
38    .page-header p
39    {
40      background-color: #f0f0c0; /*light yellow*/
41    }
42  </style>
```

```
49  <div class="page-header" id="intro">
50    <p class="greeting">
51      Hello world!
52    </p>
53  </div>
```

CSS specificity

- Style rules are ranked by **specificity.**

```
11  <style type="text/css">
12    body
13    {
14      margin: 0px;
15      padding: 10px;
16      background-color: #e0e0e0; /*light grey*/
17      color: #606060; /*dark grey*/
18    }
19
20    p
21    {
22      margin: 0px;
23      padding: 5px;
24      background-color: #c0f0c0; /*light green*/
25      color: #006600; /*dark green*/
26    }
27
28    .greeting
29    {
30      background-color: #c0c0f0; /*light blue*/
31    }
32
33    #intro p
34    {
35      background-color: #f0c0c0; /*pink*/
36    }
37
38    .page-header p
39    {
40      background-color: #f0f0c0; /*light yellow*/
41    }
42  </style>
```

```
49  <div class="page-header" id="intro">
50    <p class="greeting">
51      Hello world!
52    </p>
53  </div>
```

Hello world!

Selectors and cascading - tips

- Class vs. ID

Selectors and cascading - tips

- Class vs. ID
  - Use class when what you're styling isn't unique.

Selectors and cascading - tips

- Class vs. ID
  - Use class when what you're styling isn't unique.
  - Give the body element on each page in your application a unique ID.

Selectors and cascading - tips

- Class vs. ID
  - Use class when what you're styling isn't unique.
  - Give the body element on each page in your application a unique ID.
- Complex layouts need a lot of CSS. More CSS = more potential conflicts. How do you minimize them?

Selectors and cascading - tips

- Class vs. ID
  - Use class when what you're styling isn't unique.
  - Give the body element on each page in your application a unique ID.
- Complex layouts need a lot of CSS. More CSS = more potential conflicts. How do you minimize them?
  - Intelligent selector choice

Selectors and cascading - tips

- Class vs. ID
  - Use class when what you're styling isn't unique.
  - Give the body element on each page in your application a unique ID.
- Complex layouts need a lot of CSS. More CSS = more potential conflicts. How do you minimize them?
  - Intelligent selector choice
  - Be minimal; only bring out the big guns when you need to.

Selectors and cascading - tips

- Class vs. ID
  - Use class when what you're styling isn't unique.
  - Give the body element on each page in your application a unique ID.
- Complex layouts need a lot of CSS. More CSS = more potential conflicts. How do you minimize them?
  - Intelligent selector choice
  - Be minimal; only bring out the big guns when you need to.
  - Conflicts are inevitable. How can you deal with them? Coming up…

Mini Q&A

JavaScript

JavaScript

- Today's web requires JavaScript
  - Know when to use it, and when not to.

JavaScript

- Today's web requires JavaScript
  - Know when to use it, and when not to.
  - Read *A List Apart*, and resort to JavaScript only when you have to.

JavaScript

- Today's web requires JavaScript
  - Know when to use it, and when not to.
  - Read *A List Apart*, and resort to JavaScript only when you have to.
  - Learn the DOM.

JavaScript

- Today's web requires JavaScript
  - Know when to use it, and when not to.
  - Read *A List Apart*, and resort to JavaScript only when you have to.
  - Learn the DOM.
- From rollovers to AJAX, JS has come a long way. Frameworks enable you to do OO programming or complex animations and effects.
  - Prototype and Scriptaculous
  - JQuery
  - MooTools
  - Yahoo! UI library

JavaScript performance

- Use the frameworks; be wary of performance impact.

JavaScript performance

- Use the frameworks; be wary of performance impact.
- Most users will hit your application with an empty cache; downloading a lot of files can seriously hurt performance.

JavaScript performance

- Use the frameworks; be wary of performance impact.
- Most users will hit your application with an empty cache; downloading a lot of files can seriously hurt performance.
- Moral of the story: strike a good balance between code organization and file minimization.

JavaScript performance

- Use the frameworks; be wary of performance impact.
- Most users will hit your application with an empty cache; downloading a lot of files can seriously hurt performance.
- Moral of the story: strike a good balance between code organization and file minimization.
- Browser-specific JS performance discussion coming up…

Mini Q&A

Browser compatibility

Browser compatibility

- This is where the fun begins. If you define “fun” as eye surgery. Without anesthesia.
- Ensuring browser compatibility sucks. But you can make the pain bearable.

Browser compatibility

- This is where the fun begins. If you define “fun” as eye surgery. Without anesthesia.
- Ensuring browser compatibility sucks. But you can make the pain bearable.
  - Code to the standards first. Test with Firefox 3, Safari 3, or WebKit. Then worry about (in order): Firefox 2, Internet Explorer 7, Internet Explorer 6.

Browser compatibility

- This is where the fun begins. If you define “fun” as eye surgery. Without anesthesia.
- Ensuring browser compatibility sucks. But you can make the pain bearable.
  - Code to the standards first. Test with Firefox 3, Safari 3, or WebKit. Then worry about (in order): Firefox 2, Internet Explorer 7, Internet Explorer 6.
  - Be aware of common browser rendering issues.

Firefox 2 and `inline-block`

- Firefox 2 does not support `display: inline-block`.
- Workarounds
  - Float a block element. This can be dangerous.
  - Use `–moz-inline-box`
    - If there is content in the element, or
    - In an empty element if its height and width are specified.

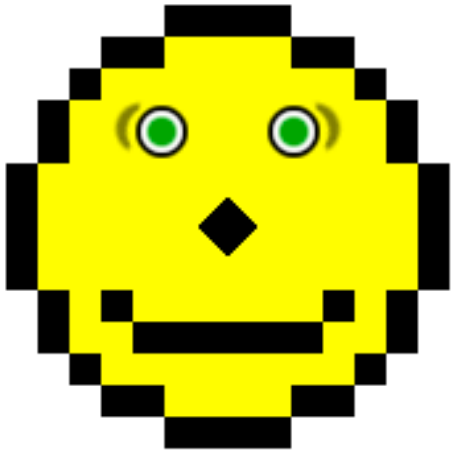
The Acid2 test

- The Acid2 test is a test page written to help browser vendors ensure proper standards support.

The Acid2 test

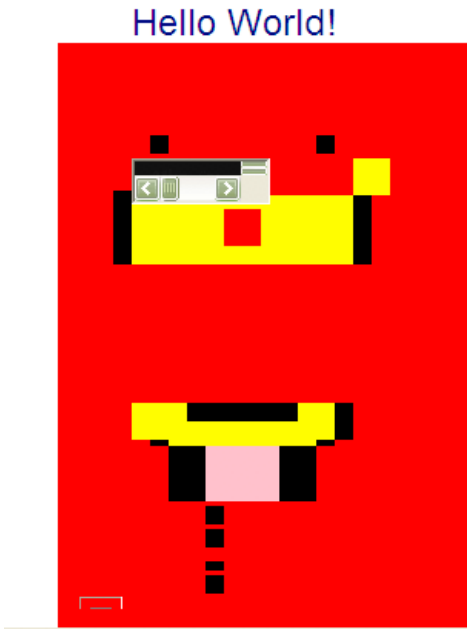
- The Acid2 test is a test page written to help browser vendors ensure proper standards support.
- How it should look:

Hello World!



The Acid2 test

- The Acid2 test is a test page written to help browser vendors ensure proper standards support.
- How it looks in IE7:

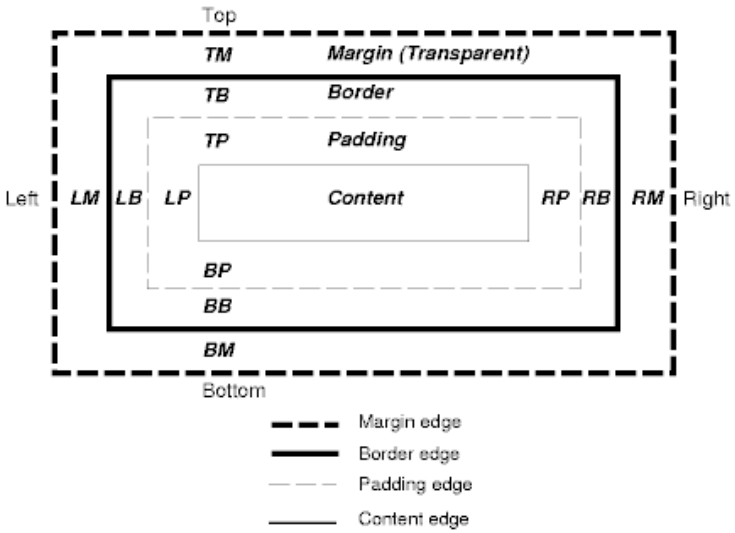


The thrill of victory; the agony of Internet Explorer

- Internet Explorer is the bane of my existence. IE7 is bad; IE6 is worse.
  - Standards support is very poor, and a slew of rendering bugs will make your life difficult indeed.
- But knowledge will help.

The thrill of victory; the agony of Internet Explorer

- Internet Explorer is the bane of my existence. IE7 is bad; IE6 is worse.
  - Standards support is very poor, and a slew of rendering bugs will make your life difficult indeed.
- But knowledge will help.
  - Understand the box model.



The thrill of victory; the agony of Internet Explorer

- Internet Explorer is the bane of my existence. IE7 is bad; IE6 is worse.
  - Standards support is very poor, and a slew of rendering bugs will make your life difficult indeed.
- But knowledge will help.
  - Understand the box model.
  - Learn how to debug IE rendering problems.

The thrill of victory; the agony of Internet Explorer

- Internet Explorer is the bane of my existence. IE7 is bad; IE6 is worse.
  - Standards support is very poor, and a slew of rendering bugs will make your life difficult indeed.
- But knowledge will help.
  - Understand the box model.
  - Learn how to debug IE rendering problems.
  - Become familiar with common bugs.

The thrill of victory; the agony of Internet Explorer

- Internet Explorer is the bane of my existence. IE7 is bad; IE6 is worse.
  - Standards support is very poor, and a slew of rendering bugs will make your life difficult indeed.
- But knowledge will help.
  - Understand the box model.
  - Learn how to debug IE rendering problems.
  - Become familiar with common bugs.
- Further reading:
  - IE7 bug reports
  - The concept of hasLayout in IE/Win

IE and the z-index bug

- The z-index bug has been around since IE 4 or 5, and has *still* not been fixed.

IE and the z-index bug

- The z-index bug has been around since IE *4* or *5*, and has *still* not been fixed.
- IE creates a new stacking context for elements with *any* z-index value, including auto.

IE and the z-index bug

- The z-index bug has been around since IE *4* or *5*, and has *still* not been fixed.
- IE creates a new stacking context for elements with *any* z-index value, including auto.
- Workaround: manually assign unique z-index values to every positioned element on your page.

IE and JavaScript performance

- IE's JS performance is abysmal.
- To mitigate slowdowns:

IE and JavaScript performance

- IE's JS performance is abysmal.
- To mitigate slowdowns:
  - Cache DOM lookups where possible[10,11,12]

IE and JavaScript performance

- IE's JS performance is abysmal.
- To mitigate slowdowns:
  - Cache DOM lookups where possible[10,11,12]
  - Take care with variable scoping

IE and JavaScript performance

- IE's JS performance is abysmal.
- To mitigate slowdowns:
  - Cache DOM lookups where possible[10,11,12]
  - Take care with variable scoping
  - Be wary of JS framework performance[13]

Mini Q&A

Wicket and markup

Wicket and markup

- CSS selectors: class vs. ID
  - Use classes on dynamic elements

Wicket and markup

- CSS selectors: class vs. ID
  - Use classes on dynamic elements
- How can I hide markup wrapping an element?
  - `setRenderBodyOnly(true);`

Wicket and markup

- CSS selectors: class vs. ID
  - Use classes on dynamic elements
- How can I hide markup wrapping an element?
  - `setRenderBodyOnly(true);`
- How can I hide the Wicket markup?
  - `getMarkupSettings().setStripWicketTags(true);`

Wicket and markup

- CSS selectors: class vs. ID
  - Use classes on dynamic elements
- How can I hide markup wrapping an element?
  - `setRenderBodyOnly(true);`
- How can I hide the Wicket markup?
  - `getMarkupSettings().setStripWicketTags(true);`
- The CSS/JS caching problem
  - One solution: append version numbers to the files dynamically
  - In Wicket: extend `MarkupParserFactory`

Coding standards and references

Coding standards

- Use spaces, not tabs
- 2 spaces/tab for HTML and CSS
- 4 spaces/tab for Java and JavaScript
- Set `svn:eol-style` to `native`[15]
- See these in action in our best practices sample project.

Tools and references

- Debugging CSS and JavaScript with only a browser is close to impossible.
- The good news: tools are available to help
  - Safari 3 and WebKit: built-in element inspector and JS console.
  - Firefox: FireBug is essential
  - Internet Explorer: the developer toolbar and MS Script Debugger

Tools and references (cont'd)

- Further reading
  - Web design reference library
  - Browser compatibility lessons learned

Conclusion

- Successful web applications treat markup with as much care as the back-end code.

Conclusion

- Successful web applications treat markup with as much care as the back-end code.
- A little time now will save you lots later.

Conclusion

- Successful web applications treat markup with as much care as the back-end code.
- A little time now will save you lots later.
- Questions?