# Implementing an HTML5 Conformance Checker Using XML Tools

Henri Sivonen

- Master's thesis project at Helsinki University of Technology
- Funded by the Mozilla Foundation

## Conformance Checker?

- Checks if the input satisfies the *machine-checkable* conformance criteria for HTML5

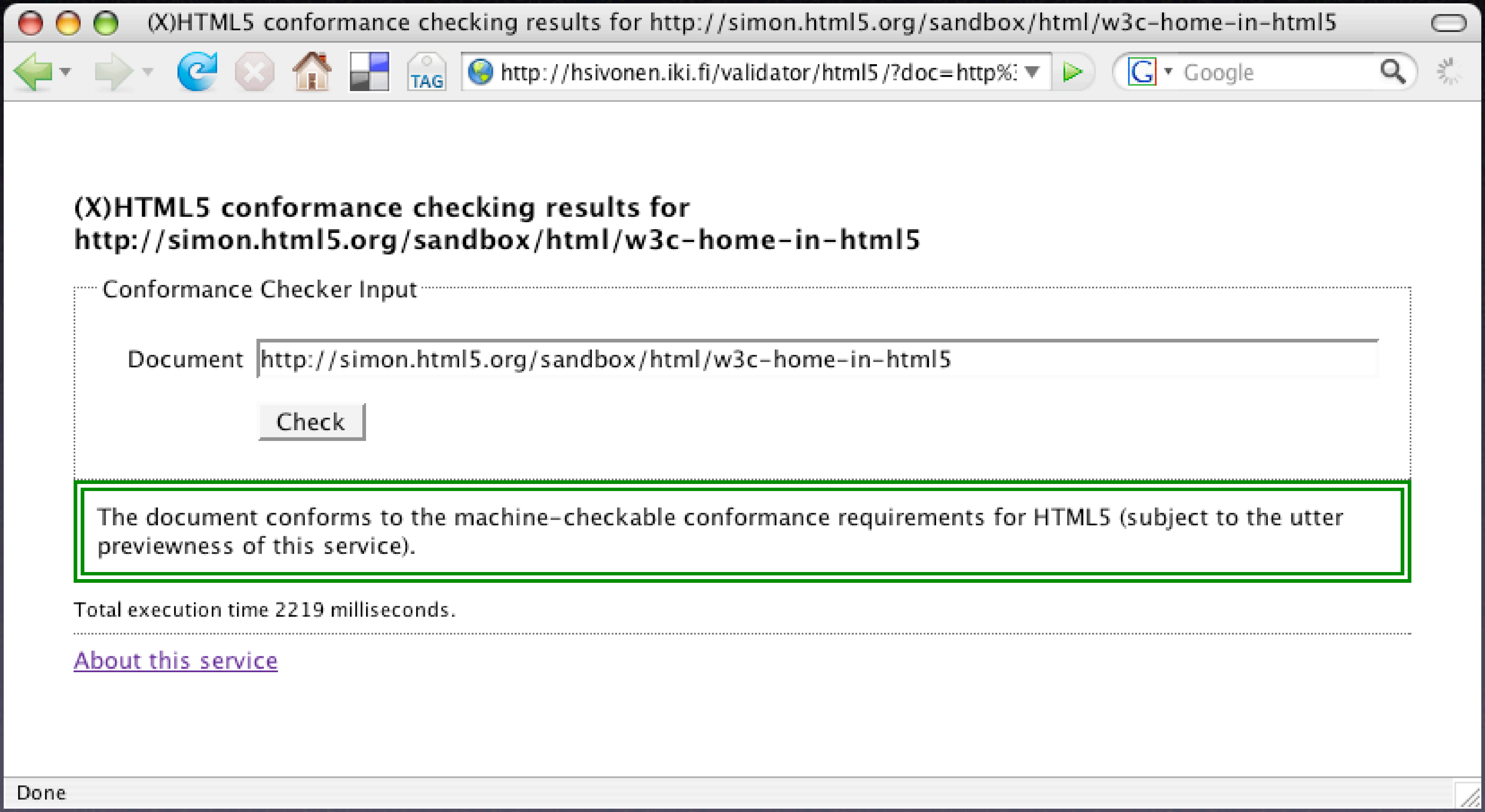
## Conformance Checker?

- Input: Document + HTTP headers
- No scripting (Halting Problem)

## Why?

- Quality assurance tool for authors
- Find errors you didn’t intend to make
- *Not* a graven image
- *No* badges

## It’s a Web App


(X)HTML5 conformance checking results for http://simon.html5.org/sandbox/html/w3c-home-in-html5
TAG
http://hsivonen.iki.fi/validator/html5/?doc=http%3
Google

(X)HTML5 conformance checking results for http://simon.html5.org/sandbox/html/w3c-home-in-html5

Conformance Checker Input

Document http://simon.html5.org/sandbox/html/w3c-home-in-html5

Check

The document conforms to the machine-checkable conformance requirements for HTML5 (subject to the utter previewness of this service).

Total execution time 2219 milliseconds.

About this service

Done

## Isn’t That a Validator?

- Colloquially: *Yes!*
- Splitting hairs: *Not exactly.*
  - Conformance requirements rule
  - Schema formalisms an impl. detail

## HTML 4 Validation

- SGML DTD-based
  - `<title/Hello/`
- Incomplete
  - `<ins datetime="blabla">` is valid … but not conforming

## HTML5 Conformance Checking

- *No* DTDs, *no* SGML parsing
- If a machine can check a requirement, *do it!*
- Schema (in)capabilities *not* an excuse
- *No* official schema
- *No* endorsed schema languages

## HTML and XML Tools

- XML has all these tools (validators, etc.)
- But HTML5 isn’t XML

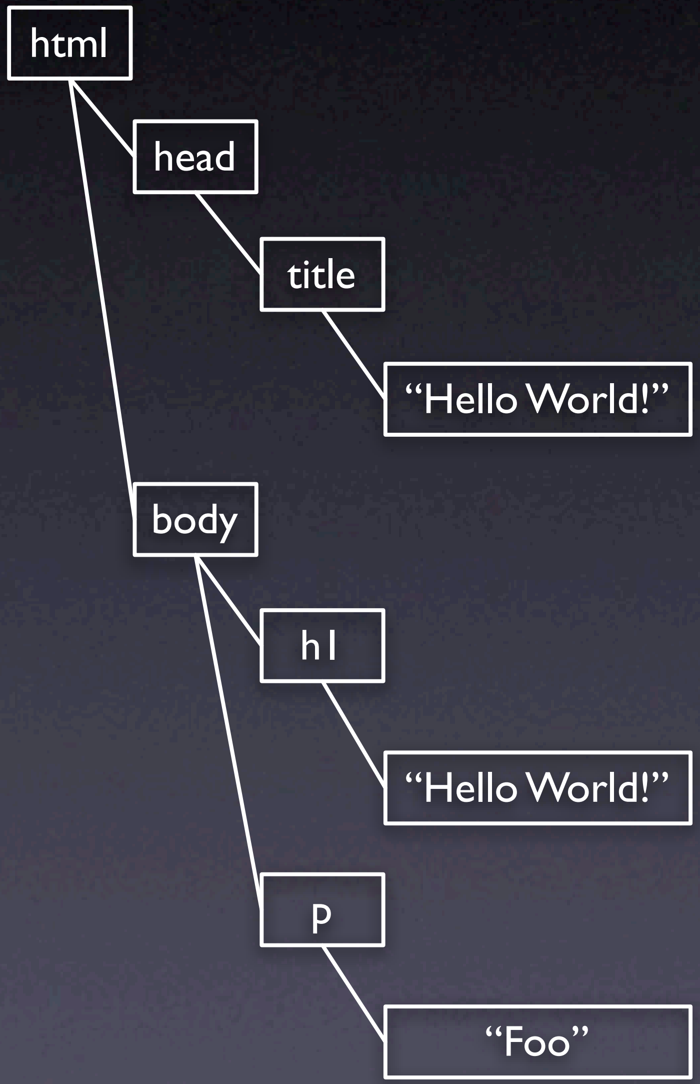
# HTML5 and XHTML5

- Two serializations
- Similar document trees
- text/html ⇒ HTML5
- application/xhtml+xml ⇒ XHTML5

# Looks Kinda Similar...

-
  ```
  <!DOCTYPE html>
  <html>
    <head>
      <title>Hello World!</title>
    </head>
    <body>
      <h1>Hello World!</h1>
      <p>Foo</p>
    </body>
  </html>
  ```

-
  ```
  <html xmlns="http://www.w3.org/1999/xhtml">
    <head>
      <title>Hello World!</title>
    </head>
    <body>
      <h1>Hello World!</h1>
      <p>Foo</p>
    </body>
  </html>
  ```



# HTML Parser

- HTML5 is almost like XHTML5

  ⇒ Map HTML5 to XHTML5 for XML tools
- Pretend to be an XML parser
- SAX interface
- Inspired by John Cowan’s TagSoup

# SAX

- Parse events as callback initiated by parser
  - `startElement`
  - `characters`
  - `endElement`

## So We Have Parsers

- XHTML5 to SAX
- HTML5 to SAX
- What’s listening to the parse events?

## Schemata?

- XSD?
- RELAX NG?
- Schematron?

## Schemata?

- ~~XSD?~~
- **RELAX NG!**
- **Schematron!**

## Enough?

- “A table model error is an error with the data represented by `table` elements and their descendants. *Documents must not have table model errors*.”
- Etc…

## No Schemata?

- Feed Validator
- Turing-complete languages can check *everything* that is machine-checkable

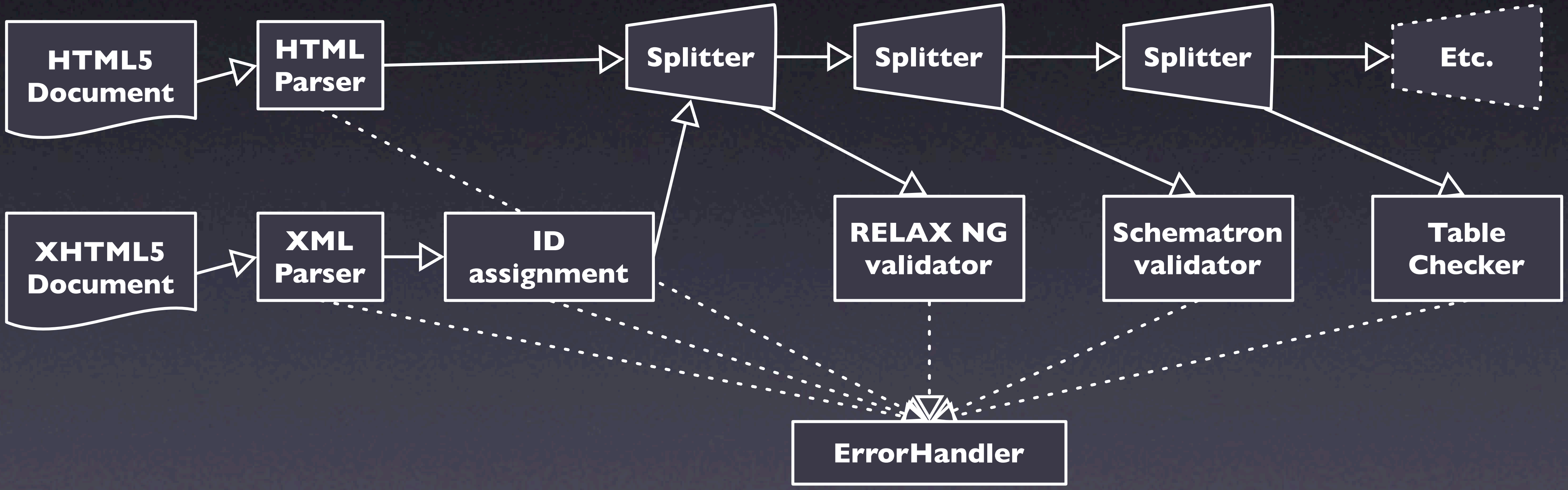
## No Schemata?

- Lots of hand-crafted code
- Wouldn’t schemata be nice as a baseline?

# Best of Both Worlds

- A RELAX NG schema as the baseline
- Refine with Schematron
- Refine even more with Java

# SAX Flow


HTML5 Document
HTML Parser
XHTML5 Document
XML Parser
ID assignment
Splitter
Splitter
Splitter
Etc.
RELAX NG validator
Schematron validator
Table Checker
ErrorHandler

# RELAX NG

- Grammar-based
- “`foo` has these attributes”
- “`foo` can have these children”
- “Attribute `bar` has this datatype (format)”
- HTML5 schema project started by fantasai

# RELAX NG

```
blockquote.elem =
  element blockquote { blockquote.inner & blockquote.attrs
                     }
blockquote.attrs =
  ( common.attrs
  & blockquote.attrs.cite?
  )
  blockquote.attrs.cite =
    attribute cite {
      common.data.uri
    }
blockquote.inner =
  ( common.inner.block )
```

# RELAX NG

```
blockquote.elem =
  element blockquote { blockquote.inner & blockquote.attrs
                     }
blockquote.attrs =
  ( common.attrs
  & blockquote.attrs.cite?
  )
  blockquote.attrs.cite =
    attribute cite {
      common.data.uri
    }
blockquote.inner =
  ( common.inner.block )
```

# Attribute Datatypes

```
blockquote.elem =
  element blockquote { blockquote.inner & blockquote.attrs
                     }
blockquote.attrs =
  ( common.attrs
  & blockquote.attrs.cite?
  )
  blockquote.attrs.cite =
    attribute cite {
      common.data.uri
    }
blockquote.inner =
  ( common.inner.block )
```

## Attribute Datatypes

```
datatypes w = "http://whattf.org/datatype-draft"

common.data.uri =
  string "" | w:iri-ref
```

# IriRef.java

```
public void checkValid(CharSequence literal) throws DatatypeException {
  IRIFactory fac = new IRIFactory();
  fac.shouldViolation(true, false);
  fac.securityViolation(true, false);
  fac.dnsViolation(true, false);
  fac.mintingViolation(false, false);
  fac.useSpecificationIRI(true);
  fac.useSchemeSpecificRules("http", true);
  fac.useSchemeSpecificRules("https", true);
  // ...
  fac.setQueryCharacterRestrictions(false);
  IRI iri;
  try {
    iri = fac.construct(literal.toString());
  } catch (IRIException e) {
    throw new DatatypeException("Bad IRI: " + e.getMessage());
  }
  try {
    iri.toASCIIString();
  } catch (MalformedURLException e) {
    throw new DatatypeException("Bad IRI: " + e.getMessage());
  }
}
```

## XSD Datatypes

- anyURI is *any string!*
- Unsuitable assumptions
- Only regular expressions useful

# Bimorphic Content Models

- HTML 4: `%Flow`, mix of block and inline
  - Arguably a DTD-induced bug
- HTML5: block or inline but not a mix
  - RELAX NG can do this
  - `<del>` complicates things a little

## Error Messages and Grammars

- Hard to explain *why* something went wrong

## Exclusions

- “`foo` cannot be a descendant of `bar`”
- Expressible in RELAX NG—in theory
- # of productions *doubles* per exclusion pair!

# Referential Integrity

- An attribute value refers to an ID
- *RELAX NG DTD Compatibility* is too naïve
  - Cannot constrain the type of referent
  - Annoying restrictions on schemata

# Schematron

-
  ```
  <rule context="h:blockquote">
    <report test="ancestor::h:header">
      The blockquote element cannot appear as a
      descendant of the header element.
    </report>
  </rule>
  ```

-
  ```
  <rule context='h:input[@list]'>
    <assert test='id(@list)/self::h:datalist or
                  id(@list)/self::h:select'>
      The list attribute of the input element must
      refer to a datalist element or to a select element.
    </assert>
  </rule>
  ```

# Schematron

-
  ```
  <rule context="h:blockquote">
    <report test="ancestor::h:header">
      The blockquote element cannot appear as a
      descendant of the header element.
    </report>
  </rule>
  ```

-
  ```
  <rule context='h:input[@list]'>
    <assert test='id(@list)/self::h:datalist or
                  id(@list)/self::h:select'>
      The list attribute of the input element must
      refer to a datalist element or to a select element.
    </assert>
  </rule>
  ```

## Java

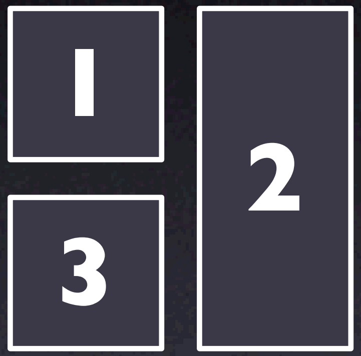
- Table integrity checker
- Unicode normalization checking
- Format of text content of elements
- Etc…

# Table Integrity

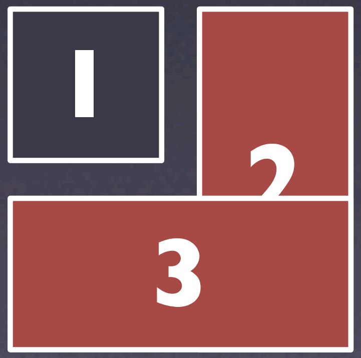
- Overlaps
- Spanning past end of row group
- Cells not matching declared columns
- Etc…

# Overlaps

-
  ```
  <table>
    <tr>
      <td>1</td><td rowspan='2'>2</td>
    </tr>
    <tr>
      <td>3</td>
    </tr>
  </table>
  ```

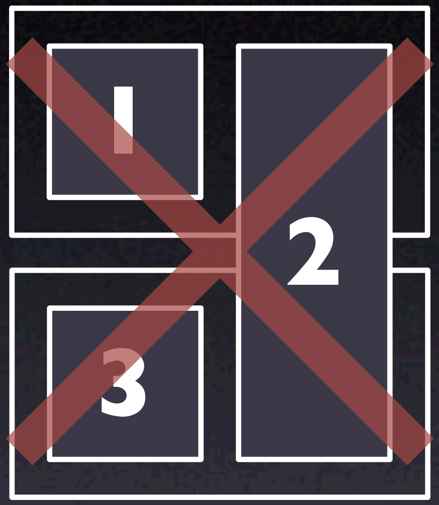


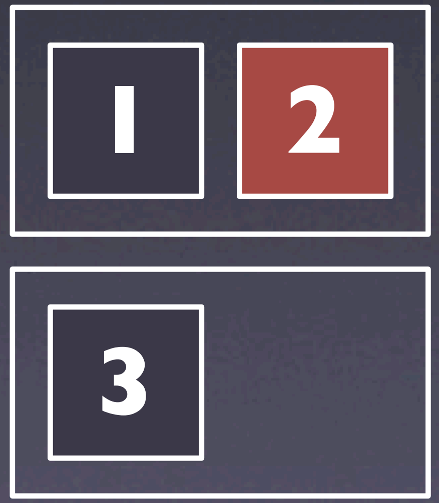
-
  ```
  <table>
    <tr>
      <td>1</td><td rowspan='2'>2</td>
    </tr>
    <tr>
      <td colspan='2'>3</td>
    </tr>
  </table>
  ```



# Spanning Past Group

-
  ```
  <table>
    <thead>
      <tr>
        <th>1</th><th rowspan='2'>2</th>
      </tr>
    </thead>
    <tbody>
      <tr>
        <td>3</td>
      </tr>
    </tbody>
  </table>
  ```





## Conclusions

## Correct Expectations

- Mapping HTML5 to XHTML5 works
- Schemata insufficient but easy to develop
- Non-schema-based checkers needed
- The quality of error messages from RELAX NG validation is a problem

## RELAX NG Surprises

- RELAX NG less applicable than expected
- Bad for exclusions
- *RELAX NG DTD Compatibility* more trouble than it is worth

## Schematron Surprises

- Less applicable than expected
  - Ancestor–descendant relationships
  - Referential integrity
- Embedding Schematron inside RELAX NG is overrated
- Could be treated as a rapid prototype

## Non-Schema

- Necessary to cover all of HTML5
- Schemata just can’t compete with the table integrity checker
- Lots of lines of code for simple things

## Questions?

- http://hsivonen.iki.fi/thesis/
- http://hsivonen.iki.fi/validator/html5/