
# openHPI Web Technologies 2015
# Week 3

**Prof. Dr. Christoph Meinel**
Hasso Plattner Institute
University of Potsdam, Germany


openHPI

# HTML Documents

**Prof. Dr. Christoph Meinel**
Hasso Plattner Institute
University of Potsdam, Germany

The WWW – World Wide Web – thrives on its contents – the linked hypermedia documents

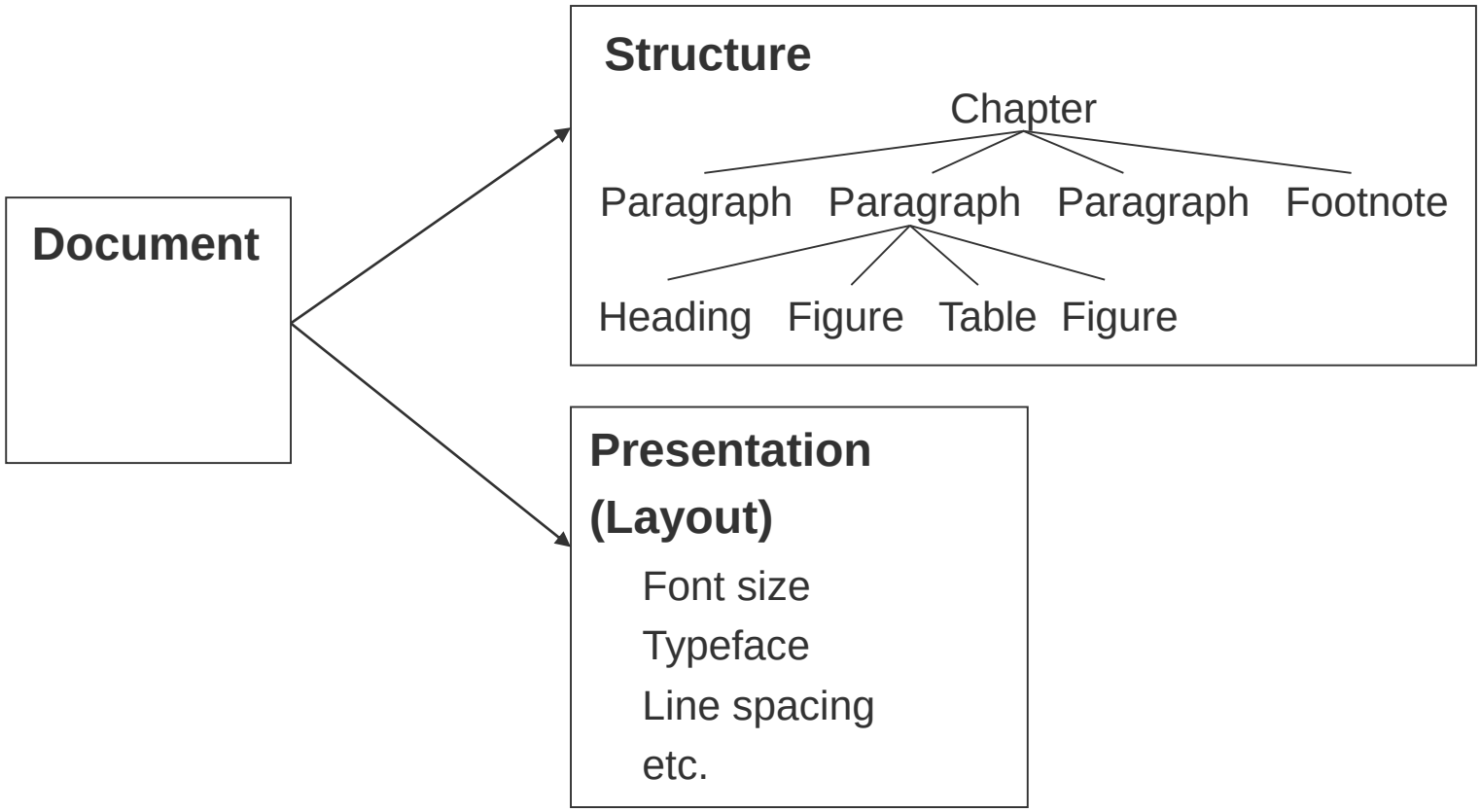
**HTML** – **Hypertext Markup Language** – is a simple markup language for the description of these hypermedia documents

- HTML is the third basic component of the WWW, besides URI and HTTP
- HTML does not indicate a format or layout-related description of the hypermedia documents, but rather describes the **structure** of their contents
- The simplicity of HTML is a major reason for the popularity of the WWW
- HTML was originally based on **SGML** – **Standard Generalized Markup Language**

# SGML (1/3)

- **SGML** – **Standard Generalized Markup Language** – was developed in the 1960s when modern data processing began
- SGML offers a meta language describing various data formats for every type of text document, independent from manufacturers, devices or applications
- **Basic idea**: Separation of
  - document **structure** – logical division and interdependency of individual document parts – and
  - document **presentation** – formatting and appearance of documents
- **Advantage**: From the single SGML document file many different layout variants can be generated for that document – printing as journal or book version, display on smartphone or pc desktops, …

## Separation of Document Structure and Document Presentation



# SGML (3/3)

- Structural elements are characterized by so-called **markups**
  - Markups are distinguished by combinations of special symbols and stored as part of the document file
    - Markup delimiter – "< … >", e.g. <chapter> … </chapter>
    - This makes it easier for the interpreter to recognize tags
- Relations between the structural elements of a SGML document are described by the SGML **document tree**
- SGML allows definition of document classes, e.g. "Book", to simplify description of the different instances of that class
  - A document class is defined by a **document type definition** – **DTD**

# Basic Structure of a HTML Document (1/5)

- **HTML** is a markup language based on SGML
- HTML is simpler than SGML because HTML is not a meta-markup language – it can be considered as one special document type …
- HTML implements the hyperlink concept of the WWW
- **HTML documents** are displayed by a browser that must additionally have access to a HTML interpreter to be able to identify the individual structural elements and assign them certain formats

# Basic Structure of a HTML Document (2/5)

- Due to historical reasons, HTML does not adhere strictly to the separation of structure and presentation in SGML and offers a range of possibilities for direct formatting instructions
- But users can / should describe HTML documents without direct formatting instructions and instead specify the necessary instructions for formatting in a separate formatting file

  → **Cascading stylesheets (CSS)**

# Basic Structure of a HTML Document (3/5)

- A **HTML document** always starts with HTML version information – strictly speaking, this is the SGML document type definition from HTML
- A HTML always consists of a header and a body
- The **header** contains information about the HTML document that does not belong to the actual document content, e.g.
  - title, author, keywords, coding, language, ...
- The **body** contains the actual contents of the HTML document
  - Basic elements of the HTML body are:
    - headings, paragraphs, lists, tables, forms, body text, special characters, images, hyperlinks

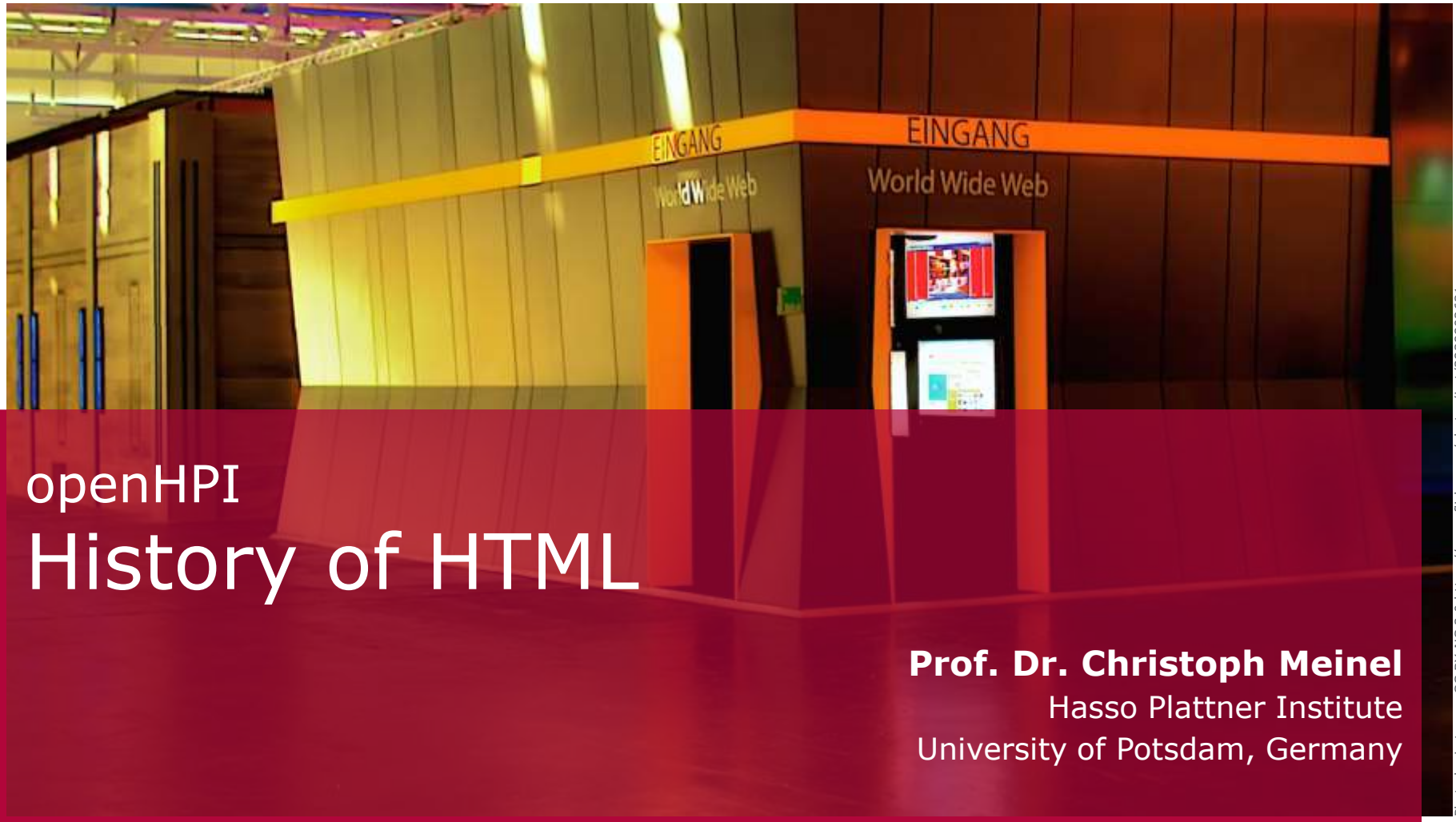
# Basic Structure of a HTML Document (4/5)

- HTML elements are identified by **markups** as in SGML
- Markups are composed of a markup identifier – **tag** –, framed by the markup delimiter "<" and ">"
- Structural elements are framed by a **start tag** and an **end tag**, e.g.  (paragraph) and 
- End tags can be left off if clarity is not lost in the interpretation
- Tags can contain additional information and control specifications in the form of **attributes**
- For HTML version 4.0, there are approx. 80 different tags specified, and for HTML 5 there are approx. 120 tags (backwards compatibility)

Hasso Plattner Institut

**Example for the Use of Attributes**

```
<div id=" CSS"><h1>Cascading Style Sheets</h1>
        <p>Text .....</p>
        <p>Text .....</p>
</div>
...
<div id="XML"><h1>Everything is Possible - XML</h1>
        <p>Text .....</p>
        <p>Text .....</p>
</div>
...
```




openHPI

# History of HTML

**Prof. Dr. Christoph Meinel**
Hasso Plattner Institute
University of Potsdam, Germany

# History of HTML (1/5)

- From the beginning, HTML was a central component in the first WWW concept (1989) introduced by Robert Cailliau and Tim Berners-Lee at the European Organization for Nuclear Research, CERN
- The first HTML version offered a hyperlink concept, but only rudimentary document description possibilities: headings in various hierarchical levels, paragraph definition, different definitions of lists
- HTML 2.0 was adopted in 1994 and went through multiple enhancements for proprietary browsers (Arena, Mosaic)
- HTML 2.0 was the first to have its own SGML-DTD

# History of HTML (2/5)

- In 1994, the **World Wide Web Consortium** – **W3C** – was founded as a standardization committee and advocacy group for WWW
- First recommendation of W3C was HTML 3.0 which, however, was only partially implemented by the competing browser manufacturers ("browser war") with the end result of being incompatible
- In 1997, HTML 2.0 was thus considered to be the only browser standard among all WWW browsers to be supported
- HTML 3.2 included many new features of individual browser manufacturers, e.g. tables, integration of Java Applets, text flow around images, and kept the lowest common denominator with respect to the simultaneous expansion of HTML 2.0

# History of HTML (3/5)

- The first proposal of **HTML 4.0** already came at the end of 1997
- The most important innovation was the use of **Cascading Stylesheets** – **CSS** – for a consistent separation between document structure and appearance
- Using CSS makes many parts of older HTML versions unnecessary, HTML 4.0 identifies these as obsolete - “deprecated”
- HTML 4.0 further supports frames for improved implantation of tables, multimedia objects and forms

# History of HTML (4/5)

**XHTML** –

- Reformulation of HTML 4.01 by virtue of XML, described as XML-DTD
- HTML had to, therefore ,be adapted to the constraints of XML, e.g. mandatory to conclude every start tag with an end tag, etc.
- Limitation of XHTML to the language elements in the Document Type Definition – Strict DTD  is standardized by W3C as XHTML 1.1
- 2002: First work on draft of XHTML 2.0

# History of HTML (5/5)

- 2004: Establishment of **WHATWG** – Web Hypertext Application Technology WG
  - members include Mozilla Foundation, Opera Software, Apple, …
  - director: Ian Hickson (author of the ACID test)
- WHATWG is working on a draft for "Web Applications 1.0", the later HTML 5, as an alternative to XHTML 2.0
- In 2006, Tim Berners-Lee published "**Reinventing HTML**", a policy paper on the development principles for **HTML 5**
- In 2009, work on XHMTL 2.0 discontinued
- W3C and WHATWG work together on HTML 5 and XHTML 5
- In May 2011, HTML 5 receives the status "last call" from W3C
- HTML 5 was completed in October 2014
- HTML 5.1 is currently a W3C "Working Draft"


openHPI

# HTML 4.0

**Prof. Dr. Christoph Meinel**
Hasso Plattner Institute
University of Potsdam, Germany

# HTML 4.0 – Document Type Definition

- The introduction of **Cascading Stylesheets** – **CSS** - made many earlier HTML constructs superfluous
- To ensure backward compatibility of earlier HTML versions, various DTDs were introduced for HTML 4.0:
  - **Transitional DTD** – intended only for interpretation of (older) HTML documents and not for generating new documents
  - **Strict DTD** – contains none of the earlier direct formatting options, used both to generate and to interpret new documents
  - **Frameset DTD** – contains only the HTML elements necessary for the definition of frames and thereby prevents recursive use of the frame construct

# Document Header and Metatags

- Header of a HTML document contains information about the particular document – **meta information** – e.g. title, name of author, key words, document description, …
- Meta information is important for the automatic evaluation of the document, e.g. for the construction of a search index for search engines
- Header is marked by tag pairs <HEAD> … </HEAD>
- The following tags can be used in the header:
  - <TITLE>, <SCRIPT>, <STYLE>, <LINK>, <META>, <NOSCRIPT>, <OBJECT>

# Document Body (1/3)

- The actual content of the HTML document is located in the **body**
- The body is enclosed by the tag pair <BODY> … </BODY>
- HTML distinguishes two categories of structural elements:
  - **Block-level elements** – define block-based structures in the document body and can contain block-based as well as inline elements
  - **Inline elements** – are in running text, can be wrapped and may contain other inline elements

# Document Body (2/3)

**Important text-based structure elements (HTML 4.0)**:

- <P> (text paragraph),
- <DIV> (division),
- <H1> to <H6> (heading),
- <SPAN> (sections in running text),
- <BLOCKQUOTE CITE="URI"> (quotes),
- <HR> (horizontal line),
- <DEL …> and <INS …> (changes)

**Lists**:

- Unordered lists – start tag: <UL>, Start tag for list entries: <LI>
- Ordered lists – start tag: <OL>, Start tag for list entries: <LI>
- Definition lists – start tag: <DL>,
  Start tag for list entries: <DT>, Start tag for list entry header: <DD>

# Document Body (3/3)

**Tables**:

- Tables consist of rectangular cells, arranged in rows and columns, using <TR> and <TD> tags
- There are many format-related attributes for tables in HTML because the CSS version 1.0 available at that time did not yet handle table styles
- Tables are framed by the tags <TABLE> … </TABLE> and consist of header <THEAD>, foot <TFOOT> and table body <TBODY> (such a division was often omitted in the past but is now making a comeback)

# Hyperlinks (1/3)

- Centerpiece of the WWW are **hyperlinks**
- Hyperlinks can link separate and independent information resources
- HTML offers different types of hyperlinks to users:
  - **invisible links**, e.g. between documents and style sheets, they are processed in the background of the browser – <LINK>
  - **visible links** which are activated by mouse click and branch off to other information resources in the Web, characterized by URIs – <A>

# Hyperlinks (2/3)

**<A>-element** can be used as a

- marker (anchor) – <A NAME="name"> … </A> –
  NAME defines a document-wide unique identifier of the anchor point and serves as a logical jump target for the browser
- jump address within the document –
  <A HREF="anchor">hyperlink text</A> –
  referenced jump target, which is fixed by the anchor tag
- jump address to an external target –
  <A HREF="URI"> hyperlink text </A> –
  branching to other information resources of the WWW
- <A>-tags can be controlled by various attributes

# Hyperlinks (3/3)

**The Future of the Hyperlinks: XLink, XPath and XPointer?**

- Different standards for the definition of links in XML documents (e.g. in XHTML)
- **XLink** should eliminate the need for <A> element: every element can become a hyperlink:
  a link
- XLink allows complex links such as multiple links
- **XPath** and **XPointer** are query languages for XML documents
- XPointer allows navigation, e.g. based on element IDs – xlink:href= "page.html#elementId" – and can replace anchor navigation
- XLink, XPath, XPointer are W3C recommendations, but not yet widely implemented (Firefox supports XLink)

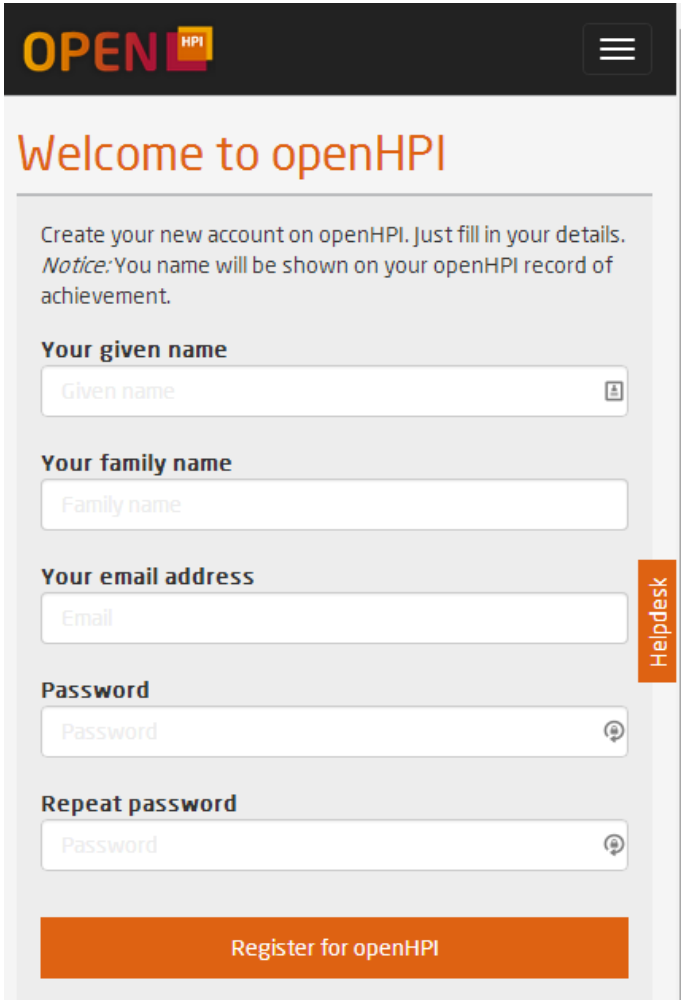
# Forms (1/2)

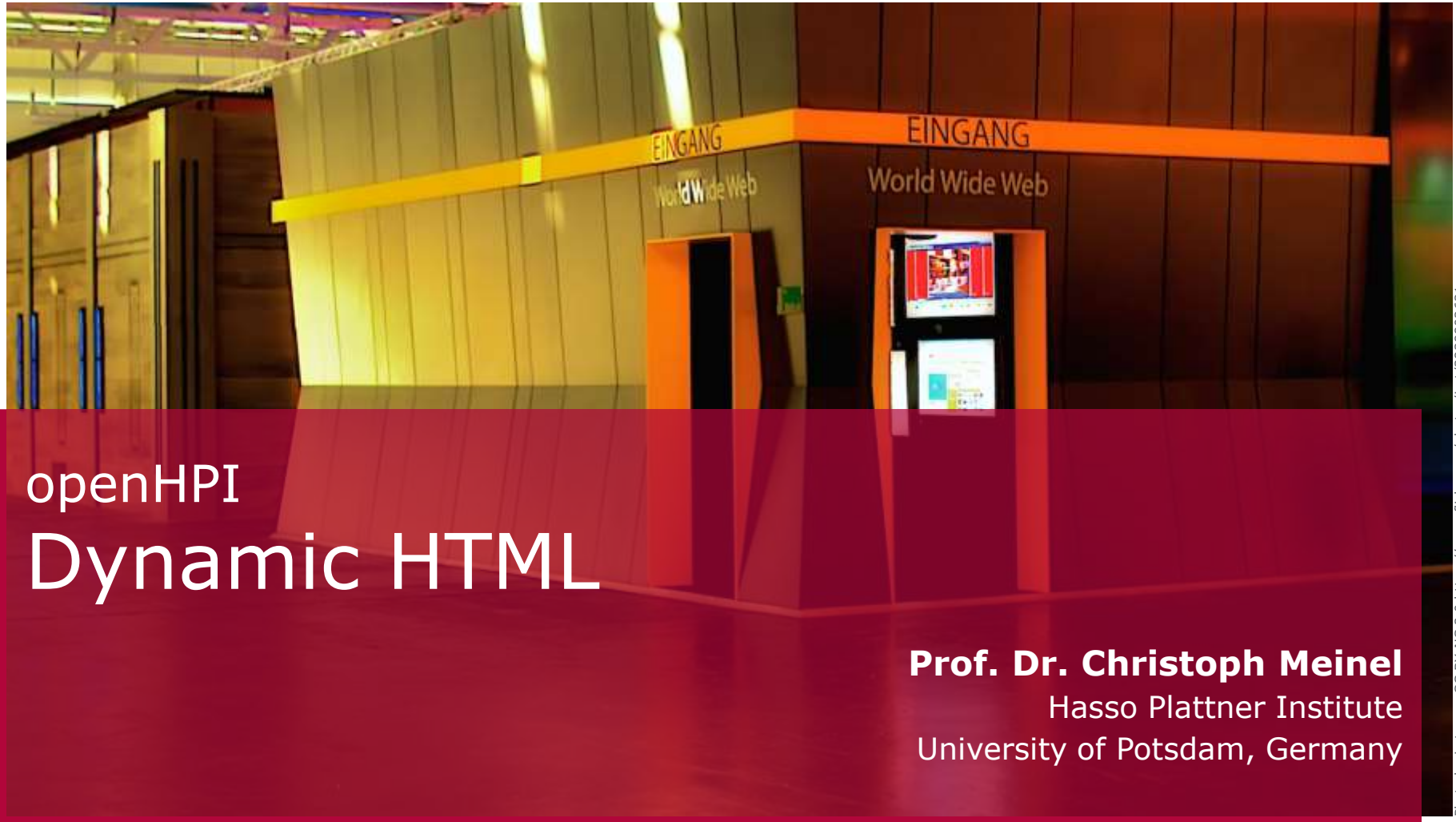
- In order to be able to send data from browser to server beyond document requirements, the concept of **forms** was introduced
- By using forms, various elements can be transmitted to the server in a data container
- Forms can be used for different tasks:
  - obtaining similarly structured (customer) information
  - specification of search requests in large datasets
  - delivery of own data to existing datasets
  - individual interaction, e.g. in online shops
  - …

# Forms (2/2)

**Example**: openHPI registration

- Data to be transmitted can be detected by the browser via input masks for entering text or selecting from list elements, and sent to the server with a mouse click
- For the specification of the form the <FORM> tag is used
- Forms consist of regular HTML elements and specific control elements – controls –, e.g. <INPUT> tag for user input






openHPI

# Dynamic HTML

**Prof. Dr. Christoph Meinel**
Hasso Plattner Institute
University of Potsdam, Germany

# Dynamic HTML

- HTML documents can capture dynamic multimedia contents and provide them to the user via the available browser
- By way of **dynamic HTML** – **DHTML**, HTML 4.0 provides client-side scripts and multimedia objects
- **Client-side scripts** are the most important component of DHTML

**Client-side scripts** are the most important component of DHTML:

- Scripts are included in the HTML document by <SCRIPT> tags
- Transferred with the HTML document they are interpreted and executed on client-side by the browser and can adapt HTML documents to user requirements
- Interaction points between HTML document and executable script are described by means of the **Document Object Model** – **DOM**
- Scripts can be executed while loading the HTML document or by the occurrence of specific events
- Client / server communication is carried out via **XMLHttpRequest**
- Formats for data transmission are e.g. **XML**, **JSON**, …

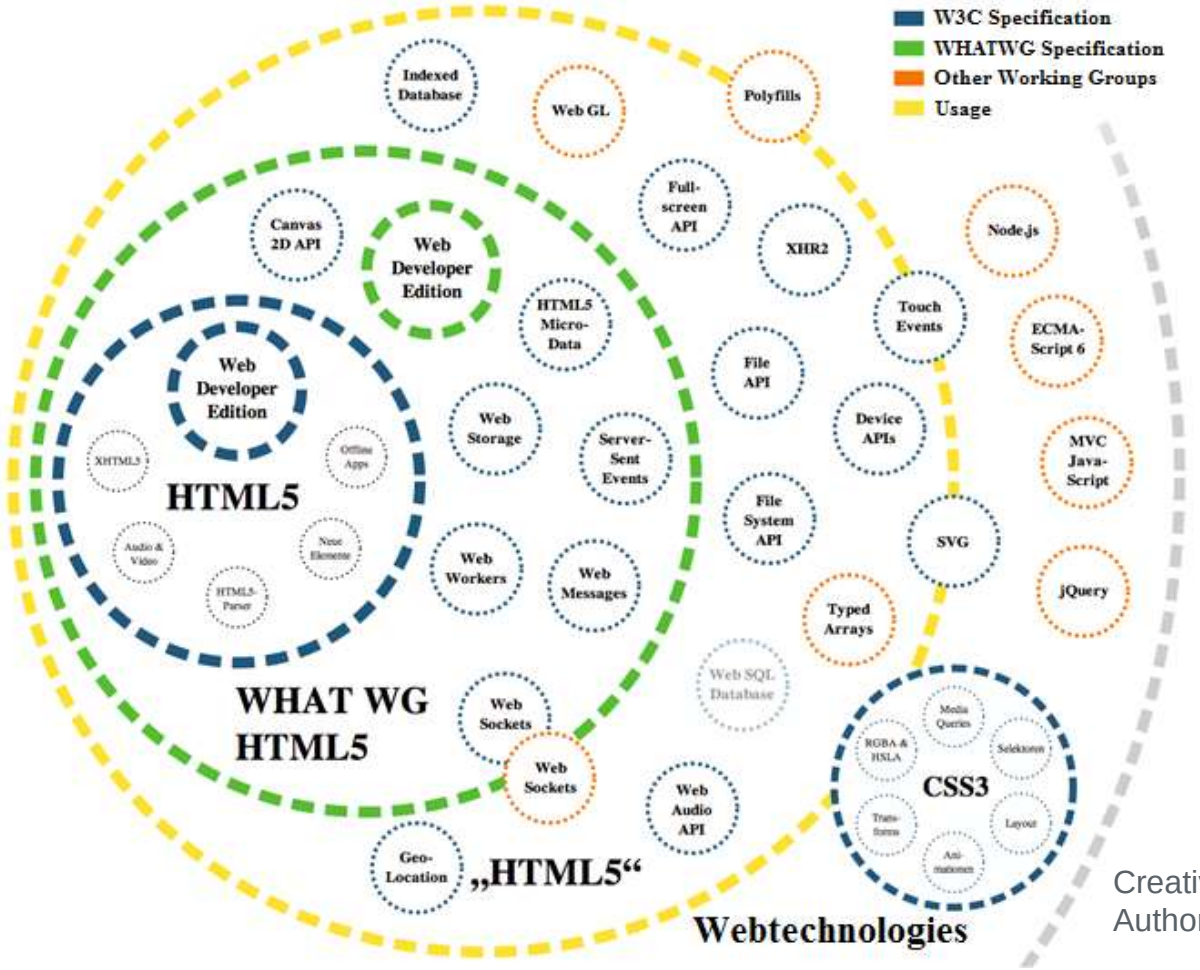
DHTML allows integration of **multimedia objects**

- Integration of any multimedia objects in HTML documents is possible within <OBJECT> environments
  - To represent these objects, the browser needs own programs or modules – **browser plugins**
  - Multimedia objects can be provided with attributes and parameters to configure execution


# openHPI
# HTML 5

**Prof. Dr. Christoph Meinel**
Hasso Plattner Institute
University of Potsdam, Germany

# HTML 5 – What is Included?




Creative Commons,
Author: Peter Kröner

# HTML 5 – Goals and Paradigms

After the HTML working group in W3C was split,
WHATWG wanted to develop a practice-oriented standard

- Standard should be jointly developed by users, authors, designers, browser manufacturers, etc.
- **Development paradigm**: "users over authors over implementers over specifiers over theoretical purity"
- HTML 5 should
  - be developed incrementally
  - have backwards compatibility
  - be easier to use
  - better implement separation between structure and layout
- HTML 5 and XHTML 5 should commonly use DOM5

# HTML 5 – What is different?

HTML 5 is no longer based on SGML

- Therefore no more DTDs: instead very simple doctype definition in the HTML header: <!doctype html>
- But: No check with existing validation tools based on DTDs
  ➔ Development of special validators for HTML 5 necessary

```
<!doctype html>
<html lang="de">
    <head>
        <meta charset="UTF-8">
        <title>HTML 5</title>
    <head>
    <body>
        <p>...</p>
    </body>
</html>
```

HTML-skeleton (minimal HTML5 document)

# HTML 5 – “Speaking” Structuring Elements (1/2)

**Introduction of “speaking” structuring elements**

- Most utilized structuring block element in HTML 4.0 has been the -tag
- HTML 5 introduces a number of elements that are to be used instead:
  - <section>
  - <article>
  - <header>, <footer>
  - <aside>
  - <nav>
  - <figure>
  - ...

# HTML 5 – "Speaking" Structuring Elements (2/2)

**Example:** "Speaking" structuring elements

```
...
<header>...</header>
<nav>...</nav>
<article id="1">
   <header>...</header>
   <section>...</section>
   <footer>...</footer>
</article>
<article id="2">
   ...
</article>
...
<aside>...</aside>
<footer>...</footer>
...
```

# HTML 5 – Multimedia Tags (1/3)

- Audio and video could previously only be used with plugins, e.g. Windows Media Player, Quicktime, VideoLAN client, etc.
- New tags <audio> and <video> allow implementation of media players directly in the browser
- Audio and video tags are of course part of the DOM
  - controls (play, pause, volume, ...) can be designed using CSS


  - Multimedia controls (jump labels, playlists, etc.) can be implemented using JavaScript

# HTML 5 – Multimedia Tags (2/3)

- Intended codecs:
  - Audio: mp3, wma, AAC, Ogg Vorbis
  - Video: H.264, Ogg Theora, WebM
- Alternative sources (different codecs or bitrates) can be provided with the <source> tag
- **Problem**: Standard does not mandate, which codec must be strictly implemented, browser manufacturers do this differently, e.g.
  - only Chrome supports all 3 codecs;
    Safari and IE only H.264;
    Firefox only Ogg Theora and WebM (Linux and Windows XP need expansion through plugins for H.264)
  - Content providers must offer multiple formats!

# HTML 5 – Multimedia Tags (3/3)

**Example**: <audio> tag with different sources

```
<audio>
   <source src="/music/audio_track.wma"
    type="audio/x-ms-wma">
   <source src="/music/audio_track.mp3"
    type="audio/mpeg">
   <p>If you can read this, your browser does not
      support the HTML audio element.</p>
</audio>
```

(Unknown tags are ignored by the browser)

# HTML 5 – Forms (1/2)

- Forms in HTML 5 are based on the Web Forms 2.0 standard
- Backwards compatibility with Web Forms 1.0 (HTML 4)
- <form> elements can be nested
- Input elements outside of a form can be associated with it by means of a form attribute:
  - <input type="text" form="my_form">
- New type and attributes for input fields allow dynamic extension of forms and client-side input validation using JavaScript
  - new <input> types: e.g.. tel, url, email, datetime, time, ...
  - new attributes: e.g. min, max, step, required, ...

**Example**: Form with validation

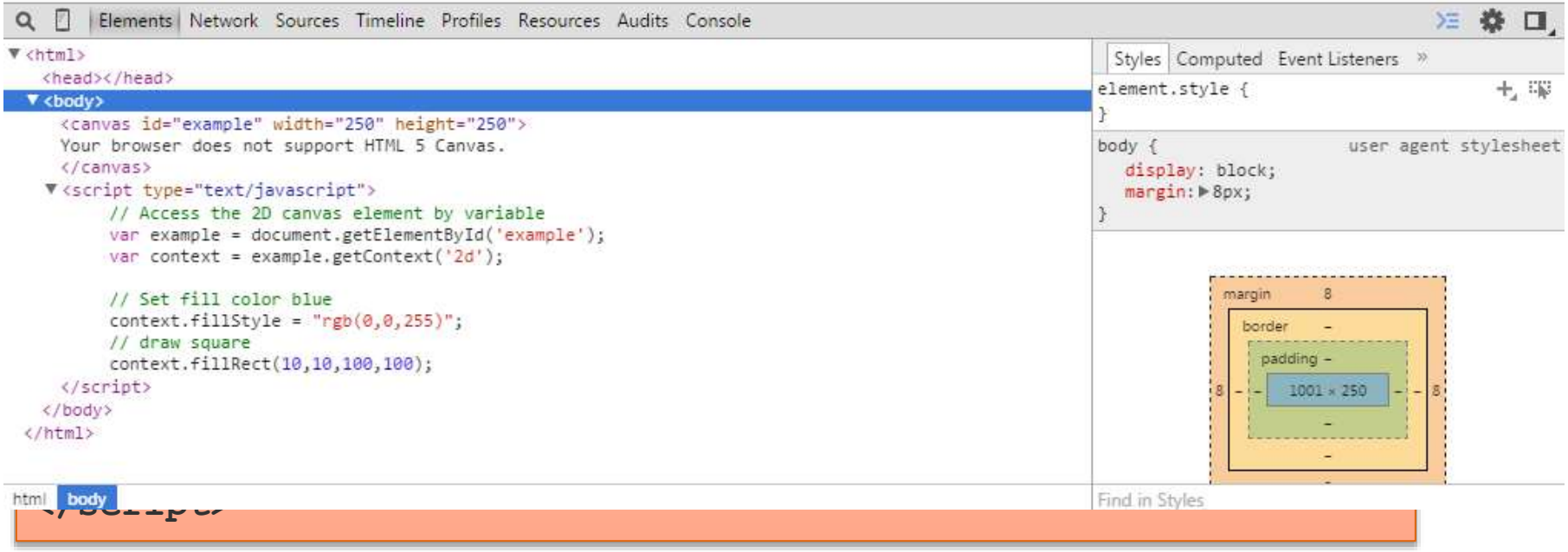
```
<form id="my_form" action="register.php" method="POST">
   Name:    <input type="text" required>
   Telefon: <input type="tel">
   E-Mail:  <input type="email" required>
   Alter:   <input type="number" min="18" required>
   <input type="submit">
</form>
...
<input type="checkbox" form="my_form"> Newsletter abonnieren
```

Note: The newsletter checkbox is outside of the form

# HTML 5 – Canvas (1/2)

- **<canvas> tag** provides workspace for bitmap graphics
- Developed originally by Apple for WebKit browser engine, now incorporated into WHATWG standard
- Drawing on a canvas with JavaScript with animation of drawings possible
- Until now, only 2D workspace, with expansion to 3D conceivable
- Supported elements are, e.g. lines, rectangles, arcs, Bézier curves or color gradients
- Canvas supports alpha channel (transparency)

**Example**: Drawing a blue square

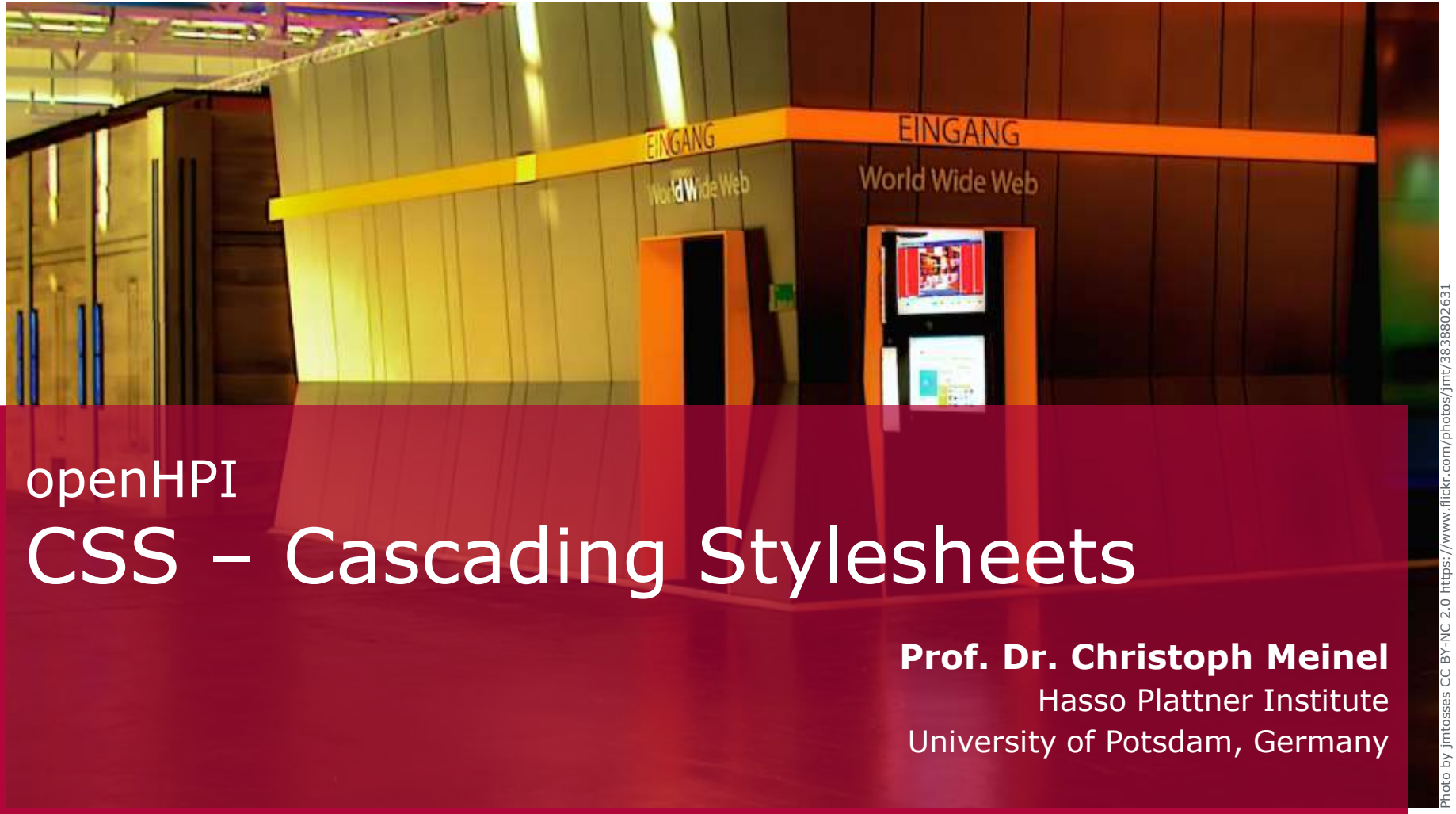


# HTML 5 – More Features

- Large number of new tags, e.g.
  - <time>, <meter> or <progress>
- Some old elements are eliminated, e.g.
  - , <font>, <u>, <frame>, <frameset>, ...
- Native support of SVG (Scalable Vector Graphics) and MathML (descriptive language for mathematical formulas)
- Integration of RDFa and microformats with HTML Microdata
- ...

# HTML 5 – Browser Support

- The latest versions of widespread browsers – Firefox, Opera, Chrome, Safari, Internet Explorer – already widely support
  - structuring tags
  - canvas
  - audio/video (but: only some of the codecs)
- HTML 5 form features (previously “Web forms 2.0”) until now
  - only extensively supported by Opera, and Chrome
  - partly supported by Internet Explorer 10+, Firefox, and Safari
- Overview browser support:
  - http://fmbip.com/litmus oder http://caniuse.com




openHPI

# CSS – Cascading Stylesheets

**Prof. Dr. Christoph Meinel**
Hasso Plattner Institute
University of Potsdam, Germany

# Introduction (1/2)

- HTML follows the basic idea of SGML: separation of document structure and document presentation
  - SGML describes the document structure
  - Formatting description language DSSSL defines the layout
- However, since at the beginning browsers did not have graphical display, layout presentation was given little attention in HTML, and initially no separate formatting description language was introduced
- Following the appearance of graphical browser interfaces (e.g. Mosaic browser 1993), the increasingly important formatting instructions were directly adopted in HTML

# Introduction (2/2)

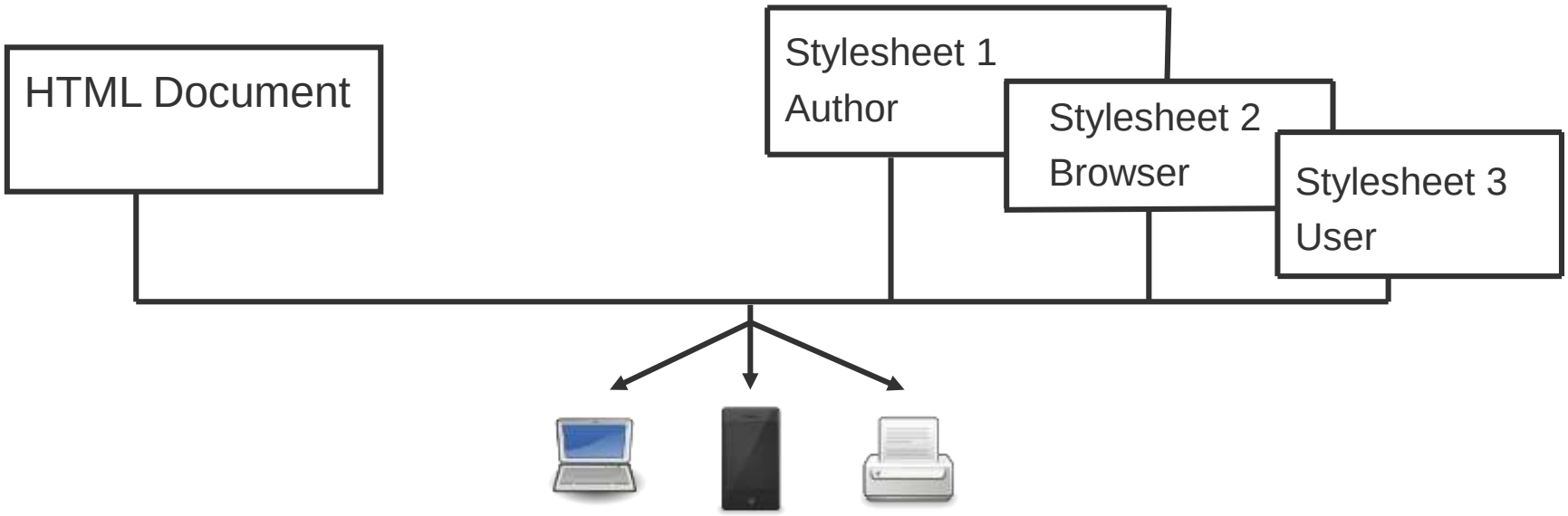
- Rapid growth of the WWW led to the desire for an appropriate document layout that is reproducible and specific for the output medium
- With **Cascading Stylesheets** – **CSS**, an independent formatting description language for describing **stylesheets** for the layout presentation of HTML documents was developed
- Whereas HTML describes the document structure, stylesheets fix the layout document presentation in special output media

# CSS – Cascading Stylesheets (1/3)

Stylesheets created with CSS allow:

- Uniform layout in related HTML documents
- Use and co-existence of entire stylesheet hierarchies that can be created by author, browser manufacturer, or user
- Design of a document according to the wishes and needs of the author and the user
- Re-use of styles across documents

**Use of Stylesheets:**



# CSS – Cascading Stylesheets (3/3)

CSS offers direct addition to HTML and assigns individual HTML commands with output-media specific layout properties

Main properties of CSS are:

- Exact definition of the layout of a HTML document
- Adaption to different output media – e.g. printer layout commands, commands for the control of artificial voice synthesis, etc.
- Central layout management - layout definition and updating can be done at a central location

# CSS – Short History

- First specification of CSS (1.0) was released in 1996 with the adoption of the HTML-4.0 standard by the W3C
- CSS 2.0 (1998) offers - besides output on different media types - print and voice output as well as pixel-accurate positioning of integrated objects
- Browser does not (yet) always work standard-compliant

  → In 2009, W3C adopted an intermediate version CSS 2.1 as "candidate recommendation" in order to correct irregularities
- Simultaneously, development was started on CSS 3, a version with modular structure. The specification phase is completed for most parts, and browser manufacturers implemented most of the W3C recommended modules.

# CSS – Connection with HTML Documents

Stylesheets can be connected with HTML documents in different ways:

- Inline definition of HTML elements with style attributes, e.g.
  - Fetter Text
- Stylesheets are defined within a HTML document using the <STYLE> tag
  - **Note**: This does not involve the strict separation of document structure and presentation
- Stylesheets are integrated from separate files using the <LINK> command in the header of the HTML document

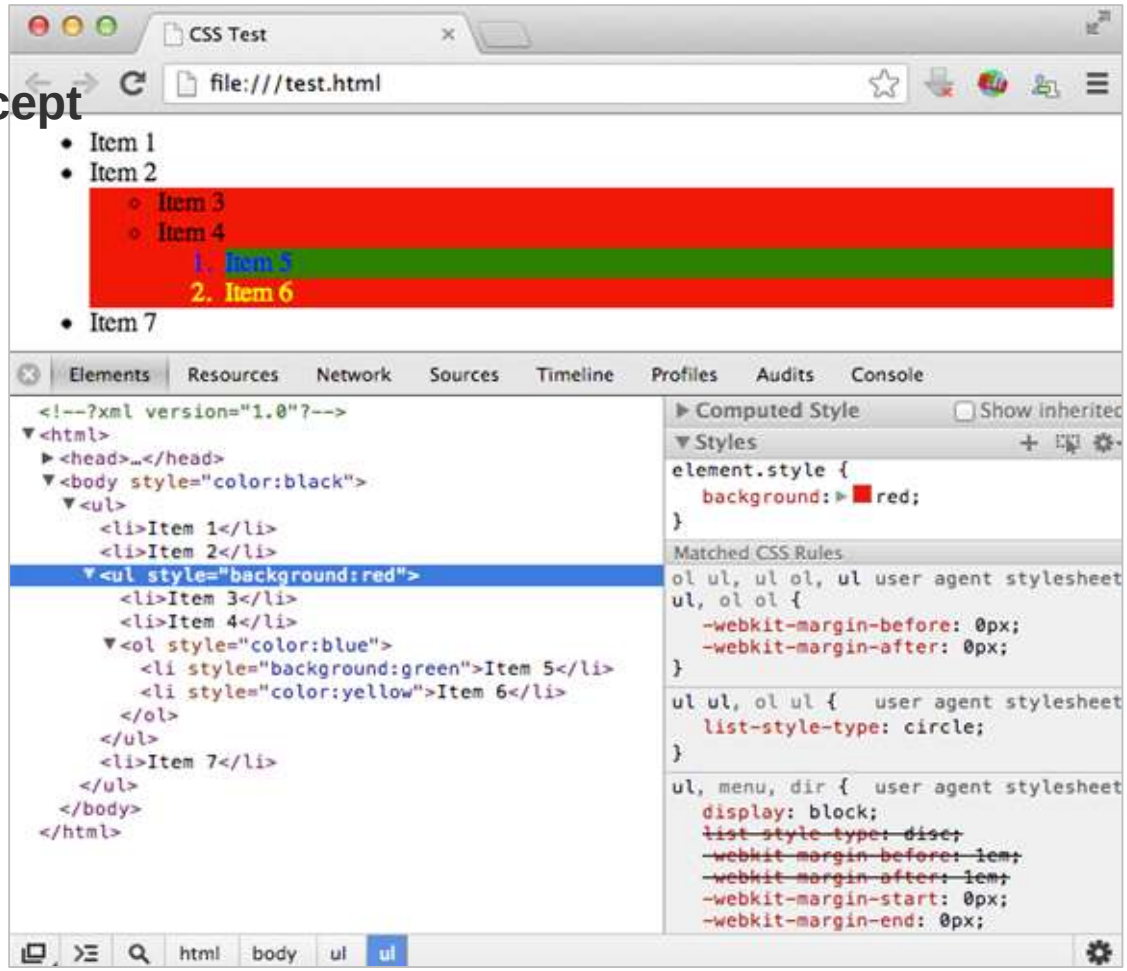
In combining both, defined styles take precedence over external, integrated stylesheets in a HTML document (local definition before global definition)

# CSS Syntax

- **CSS Stylesheet** is a collection of rules that apply to the HTML document
- **CSS rule**, e.g. `H1{color:blue}` – consists of
  - **Selector** (H1) and
  - **Declaration** ({color : blue})
- CSS formatting can be **inherited** from HTML elements in embedded elements: All HTML elements that belong to the document subtree rooted in an element inherit its formatting options

# CSS Syntax – Inheritance

Example: CSS
**Inheritance Concept**



# CSS Syntax – Declaration

**Declaration** contains actual formatting specification and assigns one or more **value**(**s**) to a **property**

Possible formatting specifications refer to the following elements:

- **Font** – font type, font size, font highlighting
- **Spacing** – based on all types of HTML elements
- **Position** – based on all types of HTML elements
- **Pictures** – based on their integration in a text
- **Colors and backgrounds** – based on all types of HTML elements

# CSS Syntax – Selector (1/3)

A **Selector** limits the potential scope of declarations (formatting instructions). There are

- **Class and ID selector** – limits the valid area to certain elements / element classes
- **Attribute selector** – formatting instructions controlled by attributes
- **Context-related selector** – limits the inheritance range of formatting instructions
- **External selector** (pseudo-elements) – applies to elements that are not part of the HTML document but part of the context

# CSS Syntax – Selector (2/3)

**Selector** limits the potential scope of declarations, e.g.:

- Standard selector **p { color:red; }**

  Use: red text

- Attribute selector **p[small] { font size:8px; }**

  Use: <p small>Das ist eine Fußnote …

  or also **p[fontsize="small"] { font-size:8px; }**
  This is a small text

- Context selector **td p { color:#0000FF; }**

  Use: <td> blue test in table </td>

  Explanation: Paragraphs (p) within table cells (td) are rendered in blue

# CSS Syntax – Selector (3/3)

**Selector** limits potential scope of declarations, e.g.:

- Class selector **p.footnote { font-family:serif; font-size:8px; }**
  This is a note
- Id selector **a#imprint { font-weight:bold; }**
  <a id="imprint" href="...">Imprint</a>
- Reminder: class attribute can be the same with multiple elements, *id* must be unique within a document!
- Class and ID selectors can also be specified without an element (and apply for different elements):

**.red { color:#FF0000; }**
**#invisible { visibility:hidden; }**

# CSS – Exact Positioning

- Prior to introduction of CSS 2.0, HTML elements could not be positioned exactly in the window screen
  → problem for web designers
- In the Netscape Navigator 4.0, a subset of CSS 1.0 was implemented with the proprietary CSS extension – **CSS-P** – for special positioning. Later, this was published by W3C
- CSS 2.0 adopted the proposed mechanism of CSS-P for exact positioning
- Furthermore, it is possible to align HTML elements relative to each other, extend them and define text flow and overlapping

**CSS 3** provides a range of other **dynamic selectors**, e.g.

- E:nth-child(…) nth child element of the element E
- E:nth-last-child(…) nth child element counted from last
- E:nth-of-type(…) nth child element of the type
- E:nth-last-of-type(…) nth child element counted from last

Instead of n other formulas are also possible, e.g.

**li:nth-of-type(2n) { color:red; }**

colors every second <li> red

# CSS – Multiple Stylesheets (1/2)

One of the basic ideas of CSS is the simultaneous use of multiple stylesheets. Possible **cascading** of the stylesheets involves:

- **Browser stylesheets** – in every document representation a browser-specific stylesheet is used
- **User stylesheets** – browsers offer users (limited) possibilities for adjusting the document representation (font, colors, etc.). Their priority is higher than browser stylesheets.
- **Author stylesheets** – authors have great freedom in layout design; they can define multiple stylesheets with
  - modular stylesheet organization
  - hierarchical stylesheet organization
  - alternative stylesheets, e.g. for different presentation forms

# CSS – Multiple Stylesheets (2/2)

Using multiple stylesheets can lead to **display conflicts** if contradictory format instructions are assigned to one HTML element – even possible within one stylesheet

CSS defines a set of rules for **conflict resolution**:

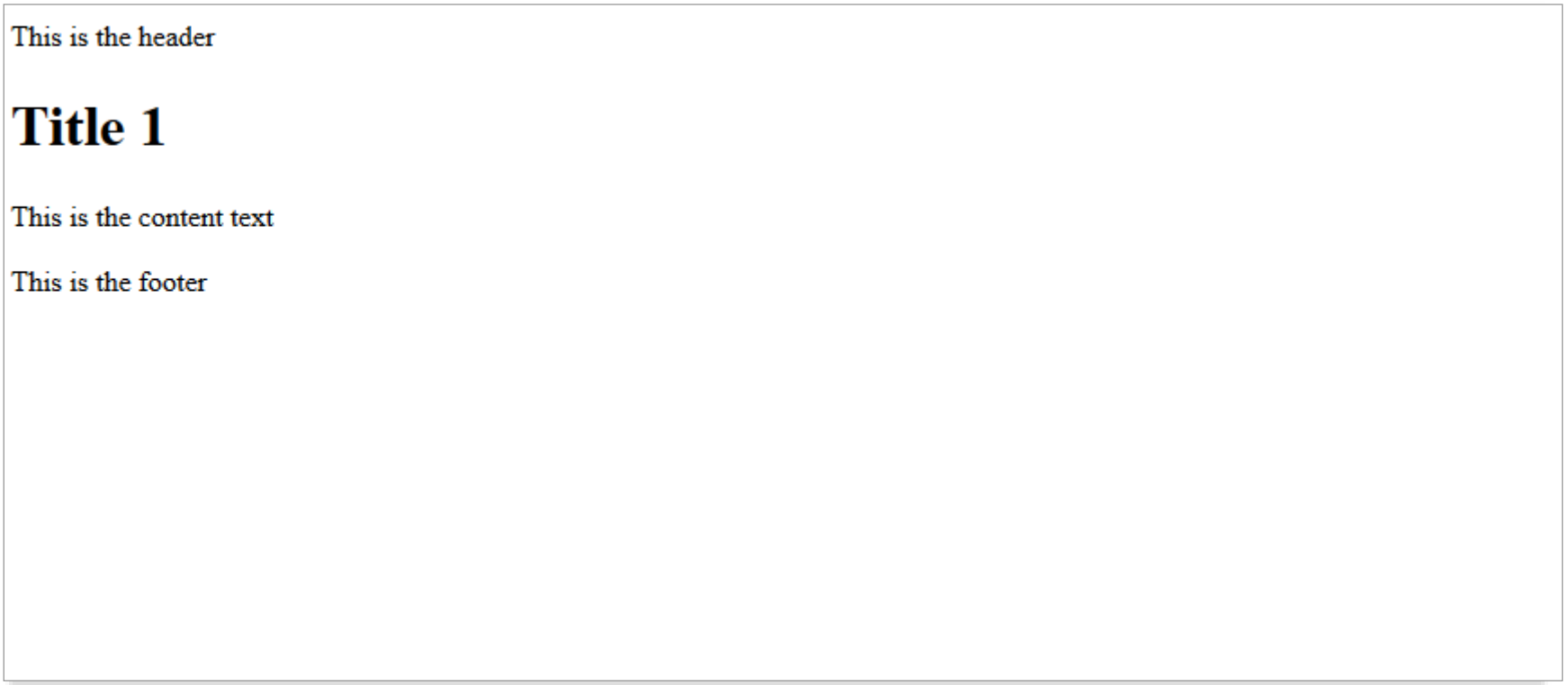
1. Find all relevant rules
2. Sort rules according to their importance – author can give rules in CSS definition the attribute "important"
3. Sort rules according to their origin: Authors before users before browser stylesheets
4. Sort rules according to their degree of specialization: Number of ID and CLASS attribute
5. Sort remaining competitive rules chronologically according to their occurrence


openHPI

# Bringing HTML and CSS Together

**Prof. Dr. Christoph Meinel**
Hasso-Plattner-Institute
University of Potsdam, Germany

## HTML Data Structure



# Insert external CSS Stylesheet File

Link HTML document with external CSS file

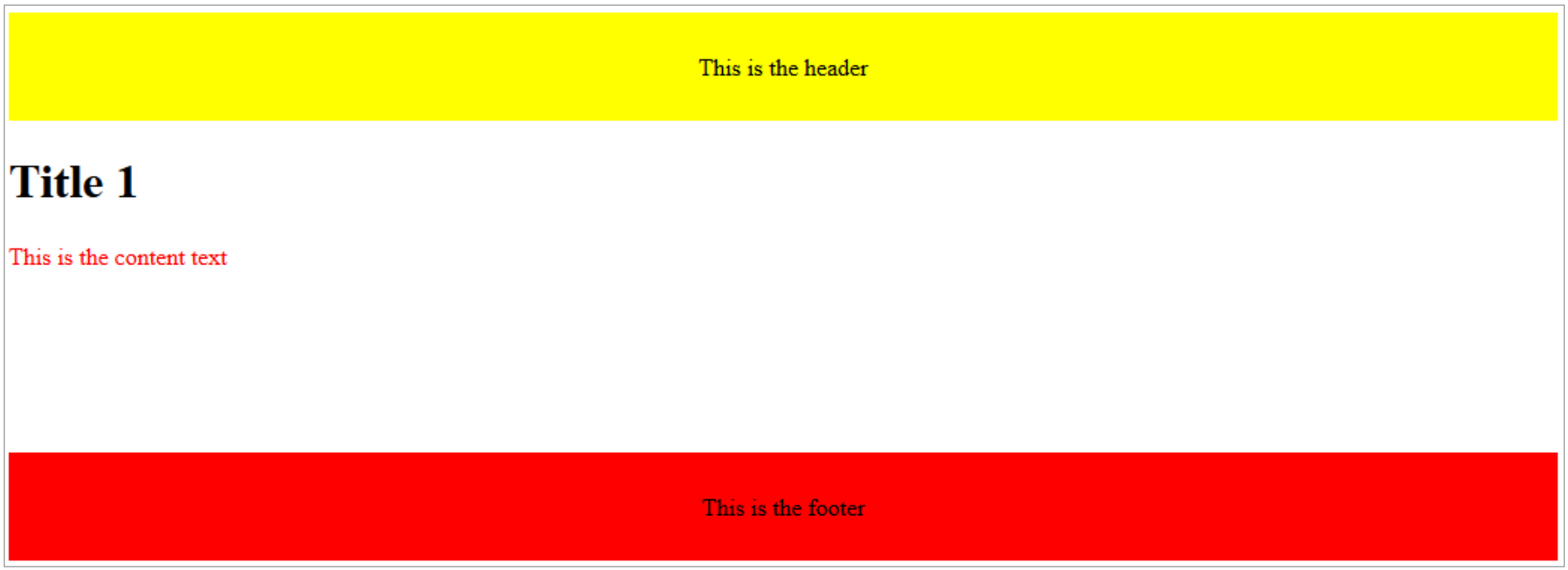
```
<html>
  <head>
    <link rel="stylesheet" type="text/css" href="style.css">
  </head>
  <body>
    <div id="header">
      <p>This is the header<p>
    </div>
    <div id="content">
      <h1>Title 1</h1>
      <p>This is the content text</p>
    </div>
    <div id="footer">
      <p>This is the footer</p>
    </div>
  </body>
</html>
```

# Write External CSS File

## CSS Style Sheet document (style.css)

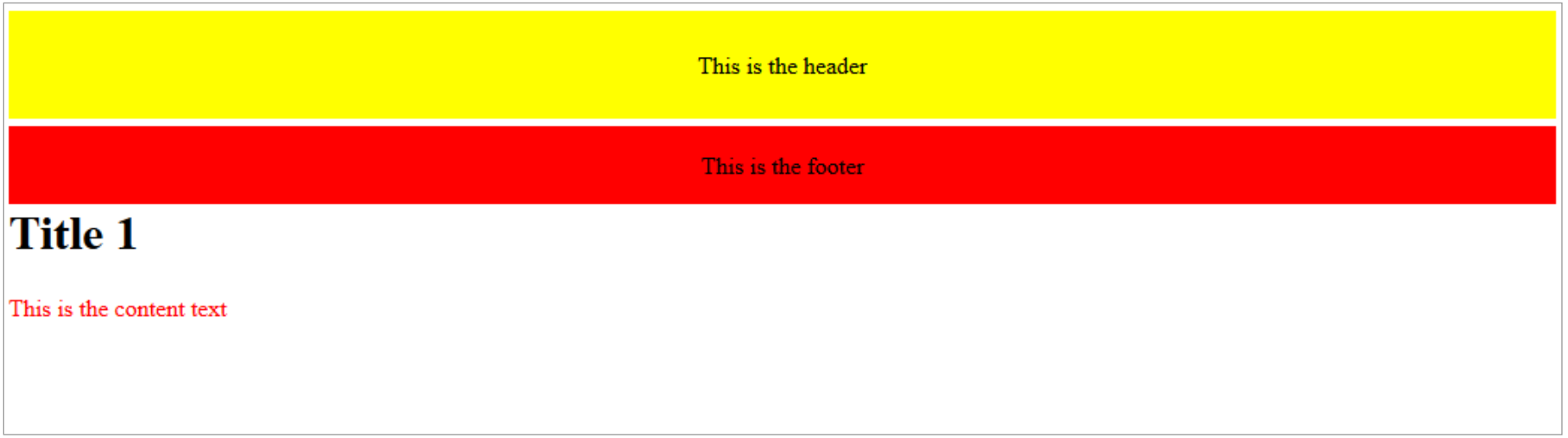
```
#header{
  background-color: #FFFF00; /*yellow*/
  padding: 10px;
  text-align: center;
}

#content{
  height: 200px;
}

#content p{
  color: #FF0000; /*red*/
}

#footer{
  background-color: #FF0000; /*red*/
  padding: 10px;
  text-align: center;
}
```

# Execute HTML Document with Linked CSS File



# Change HTML Order by Stylesheet

Manipulate footer position by CSS Style




openHPI

# XML – Extensible Markup Language

**Prof. Dr. Christoph Meinel**
Hasso Plattner Institute
University of Potsdam, Germany

# Introduction (1/2)

- A major criticism of HTML has always been its lack of flexibility
- The reason for this is the defined syntax of HTML in the **HTML Document Type Definition** (DTD)
- Authors of HTML documents cannot assign HTML elements a different meaning or define new elements themselves - a major hindrance for interaction with external applications
- This is where the **Extensible Markup Language** – **XML** – comes in. It allows the introduction of new and meaningful elements by defining own DTDs

# Introduction (2/2)

- Model for the development of XML was, like HTML, the complex meta language SGML – Standard Generalized Markup Language, with which any document could be distinguished and structured
- Since XML was designed primarily for applications on the Web, only a subset of SGML was selected to define XML in order to keep the interpretation of XML documents, authoring tools, browsers etc. as easy as possible
- In XML, only the absolutely necessary SGML provisions were adopted in order to tap the full SGML flexibility for the WWW

# XML – Extensible Markup Language (1/2)

- **XML specification** only describes the structure of documents
- **XML document** that conforms to the XML generation rules [1] is also syntactically correct, e.g.:
  - Document has exactly one root element
  - Elements have start and end tag (“<xy></xy>“, abbreviation “<xy />“)
  - ...
- XML document that conforms to the general XML specification and to the rules and definitions established in the grammar is called **valid**
- Definition of the grammar can be done, for example, by the **XML scheme** or **DTD**

[1] http://www.w3.org/TR/REC-xml/#sec-well-formed

**XML**

- Does not replace HTML, but complements it in areas where HTML has weaknesses, such as in semantic support of interaction with external applications
- General object model allows developing applications with mechanized access and the manipulation of data
  → **Document Object Model** – **DOM**
- Open standard for describing arbitrary data structures, which simplifies the exchange via the WWW and has made an integrative link between document and application

# XSL – Extensible Stylesheet Language

- XML only describes structure; determining the formatting of a document is done through the document's format description language in the form of stylesheets similar to HTML
- **Extensible Stylesheet Language – XSL** (actually XSL-FO Formatting Objects) serves in describing the layout of XML documents
- XSL is often used in close interaction with **XSL-Transformation – XSLT**
- Typical use:
  - XML document is transformed by XSLT into a XSL document, to be rendered by the FO processor into a readable / printable document

# DTD – Document Type Definition (1/3)

- XML provides authors of XML documents the possibility of a tailor-made definition of their document via their own document type descriptions - **Document Type Definitions** – **DTDs**
- To check the **validity** of XML documents, a grammar is always required, e.g. defined as DTD or XML scheme
- An **XML Parser** is required to interpret XML documents to the associated DTD – the declaration and the rules of the XML syntax
- XML Schemes are more complex but offering inheritance, XML parsing, more detailed description, …

# DTD – Document Type Definition (2/3)

Example of a DTD – **newspaper**

```
<!DOCTYPE newspaper [
        <!ELEMENT newspaper (article+)>
        <!ELEMENT article (title, introduction, text)>
        <!ELEMENT title (#PCDATA)>
        <!ELEMENT introduction (#PCDATA)>
        <!ELEMENT text (#PCDATA)>

        <!ATTLIST article author CDATA #REQUIRED>
        <!ATTLIST article lecturer CDATA #IMPLIED>
        <!ATTLIST article date CDATA #IMPLIED>

        <!ENTITY newspaper "New York Times">
        <!ENTITY publisher "Arthur Ochs Sulzberger Jr.">
]>
```

Note: Entities allow the definition of constants (special characters or frequently used text parts)

# DTD – Document Type Definition (3/3)

Example of a XML document according to the **newspaper.dtd** (cont.)

```
<?xml version="1.0" ?>
<!DOCTYPE newspaper SYSTEM "./newspaper.dtd">
<newspaper>
   <article author="Hasso Plattner" date="2015-01-07">
     <title>User Oriented</title>
     <introduction>Introduction text …</introduction>
     <text>Main article text</text>
   </article>
   <article author="Christoph Meinel" date="2015-01-09">
     <title>New Student Welcome</title>
     <introduction>Introduction text …</introduction>
     <text>Main article text</text>
   </article>
</newspaper>
```

Valid XML document with respect to a newspaper.dtd

# XML – Linking Concept (1/2)

The HTML linking concept via the hyperlink element <A> is very limited:

- HTML hyperlink links only two resources
- HTML hyperlink is always unidirectional
- HTML hyperlink is a part of the HTML from where the link originates
- HTML hyperlink does not affect the representation of the link target in the browser

# XML – Linking Concept (2/2)

Two separate language standards were developed for XML to generalize the HTML link concept and for defining links between information resources:

- **XML Linking Language** – **XLink**
  - also defines multidirectional connections
- **XML Pointer Language** – **XPointer**
  - extends the XPath specification to refer to parts of XML documents
  - **Example:**
    - xlink:href="cds.xml#xpointer(/ABBA/Waterloo)"

# Area-specific Descriptive Languages

With the XML meta language concept for defining own DTD, the path to the development of area specific descriptive languages is free, e.g.

- Mathematical Markup Language – **MathML**
- Chemical Markup Language – **CML**
- Synchronized Multimedia Integration Language – **SMIL**
- Scalable Vector Graphics – **SVG**
- …

As XML is more of a tool for specialists, the end user is only able to come in contact with XML via different applications and with XHTML as XML-conform HTML