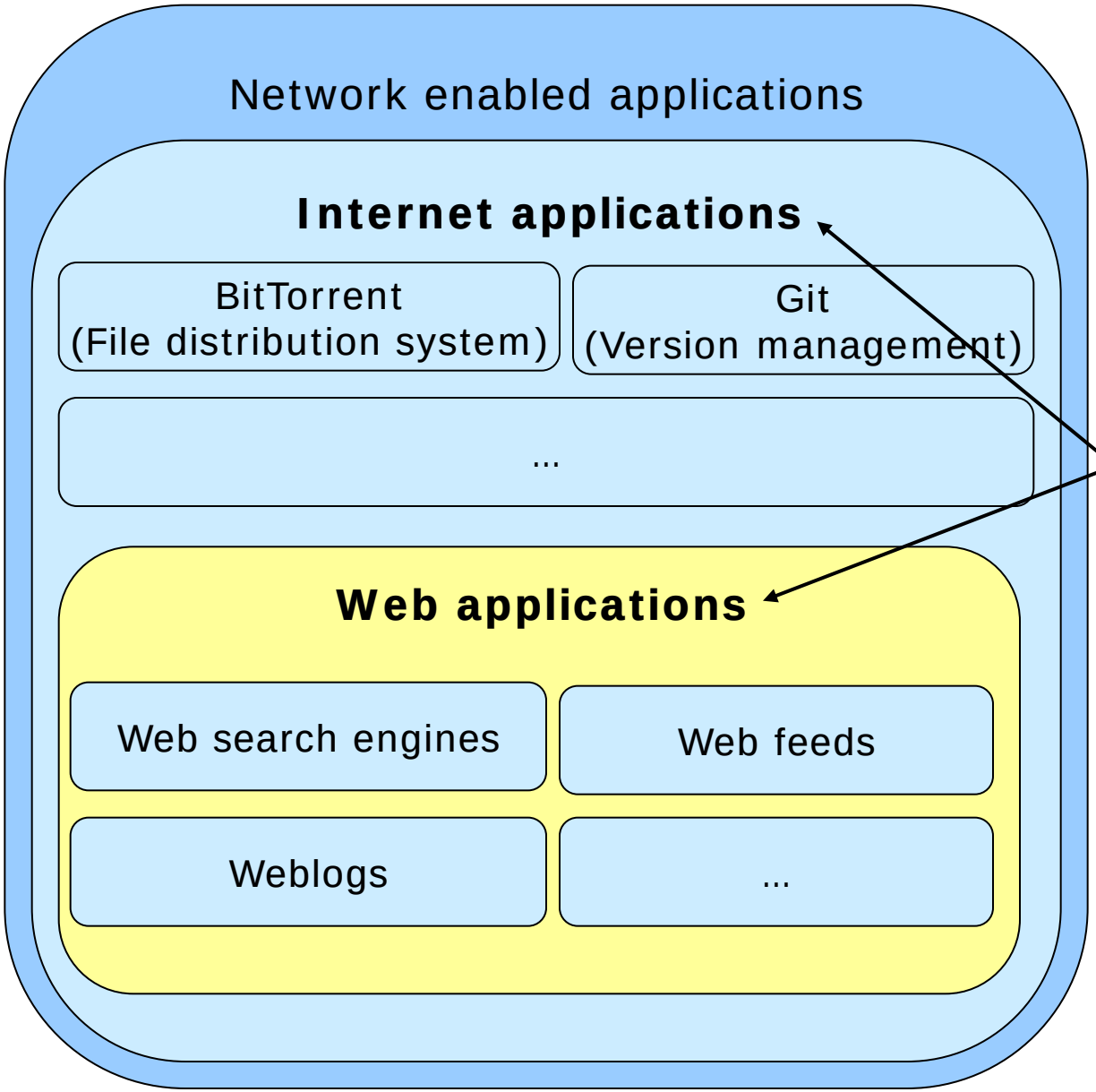
TECHNISCHE UNIVERSITÄT DRESDEN

Department of Computer Science Institute for System Architecture, Chair for Computer Networks

Internet and Web Applications

Introduction

Content



Subset of important
- applications,
- mechanisms,
- protocols and
- languages

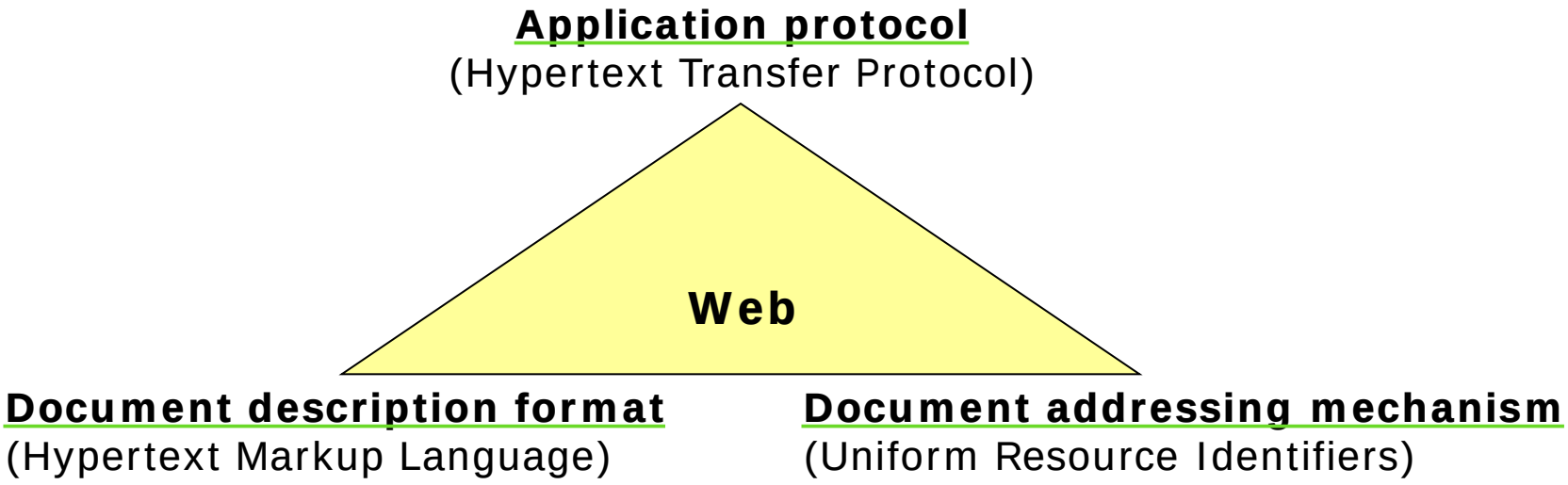
that are associated with these two categories will be discussed in the lecture

Content

1. Introduction
2. Basic aspects of Web applications
3. Interaction models in the World Wide Web
4. Extensible Stylesheet Language / Cascading Style Sheets
5. Semantic Web
6. Web Application Frameworks
7. Subscription Services
8. Web Crawling and Web Search
9. Content and File Management, Wikis
10. File Sharing
11. Load Balancing and Content Distribution
12. Business Models

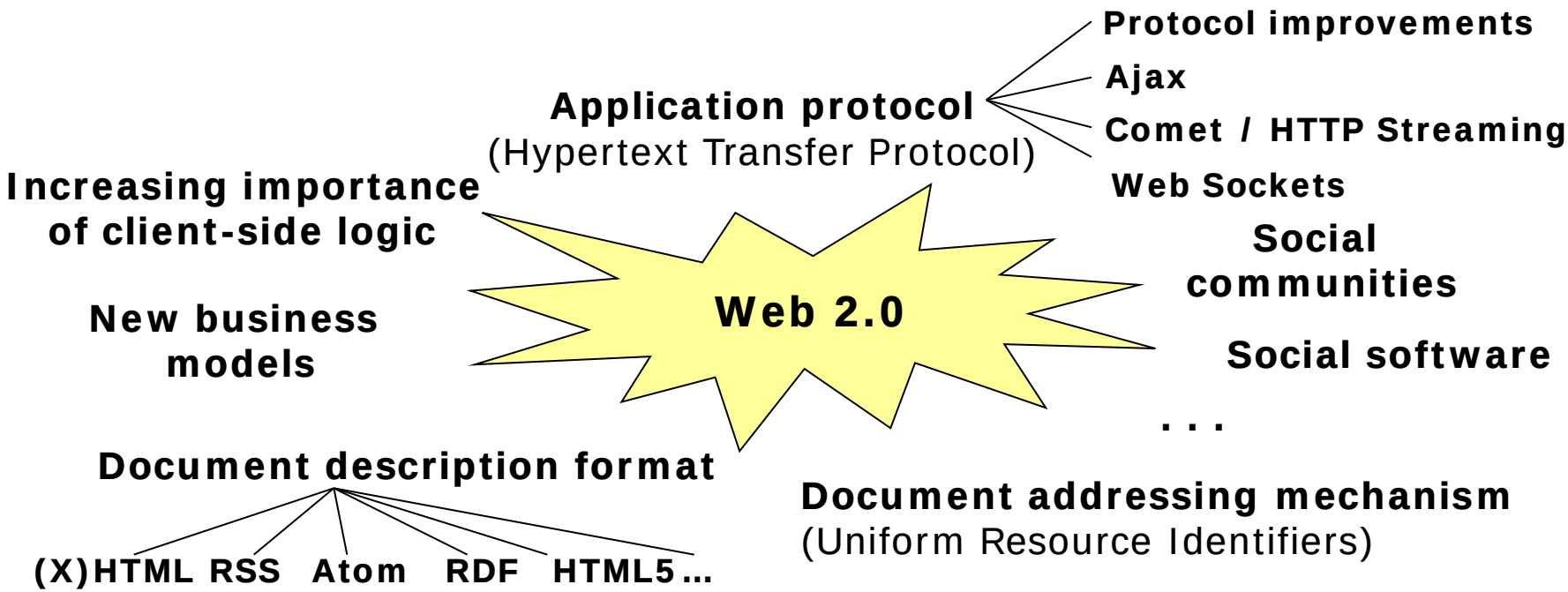
Web

- The World Wide Web is a system of hypertext documents that are viewable by a web browser
- Hypertext documents are documents with internal cross-references to other documents
- Runs as service/application on top of the Internet protocols
- Originally based on three fundamental mechanisms:



Web 2.0

- Web 2.0 is a **buzzword** for describing modern forms of applications and services that are accessible by the web browser such as wikis or weblogs
- These applications often have a high social component and appealing user interfaces in common and are based on enhanced interaction models and new document representations discussed in this lecture



Web 3.0

- Web 3.0 or the *Semantic Web* is a **vision** of *semantically* interconnected data in the Web
- While Web 2.0 has been a technological evolution, Web 3.0 is a conceptual evolution of the Web based on the enhanced technological environment
- Focus is not set to documents but to **data** thus becoming **Linked Data**
- Important formats are: RDF, RDFS, OWL (see chapter 5) as well as Microformats

> *"I have a dream for the Web [in which computers] become capable of analyzing all the data on the Web – the content, links, and transactions between people and computers. A 'Semantic Web', which should make this possible, has yet to emerge, but when it does, the day-to-day mechanisms of trade, bureaucracy and our daily lives will be handled by machines talking to machines. The 'intelligent agents' people have touted for ages will finally materialize."*
> *(Tim Berners-Lee, inventor of the original World Wide Web in Weaving the Web* ISBN 0-7528-2090-7)

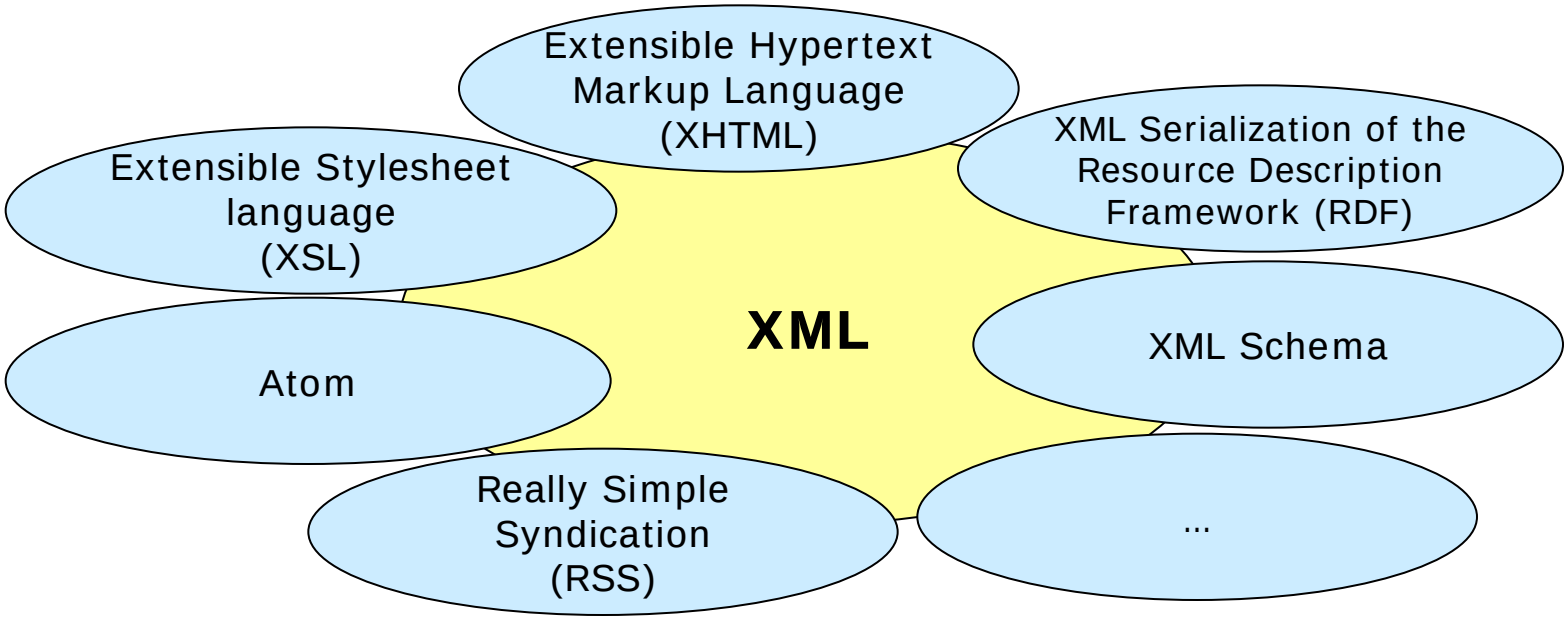
Basic knowledge

Outline

- Extensible Markup Language (**XML**)
  - Language for describing, processing and exchanging data
- Extensible Hypertext Markup Language (**XHTML**)
  - XML reformulation of HTML
- XML validation languages
  - Document Type Definition (**DTD**)
  - **XML Schema**
- XML Processing Models
  - Document Object Model (**DOM**)
  - Simple API for XML (**SAX**)
- Selected alternatives to XML
- Mechanism for message content type description
- **JavaScript**
- Cascading Style Sheets (**CSS**)

XML

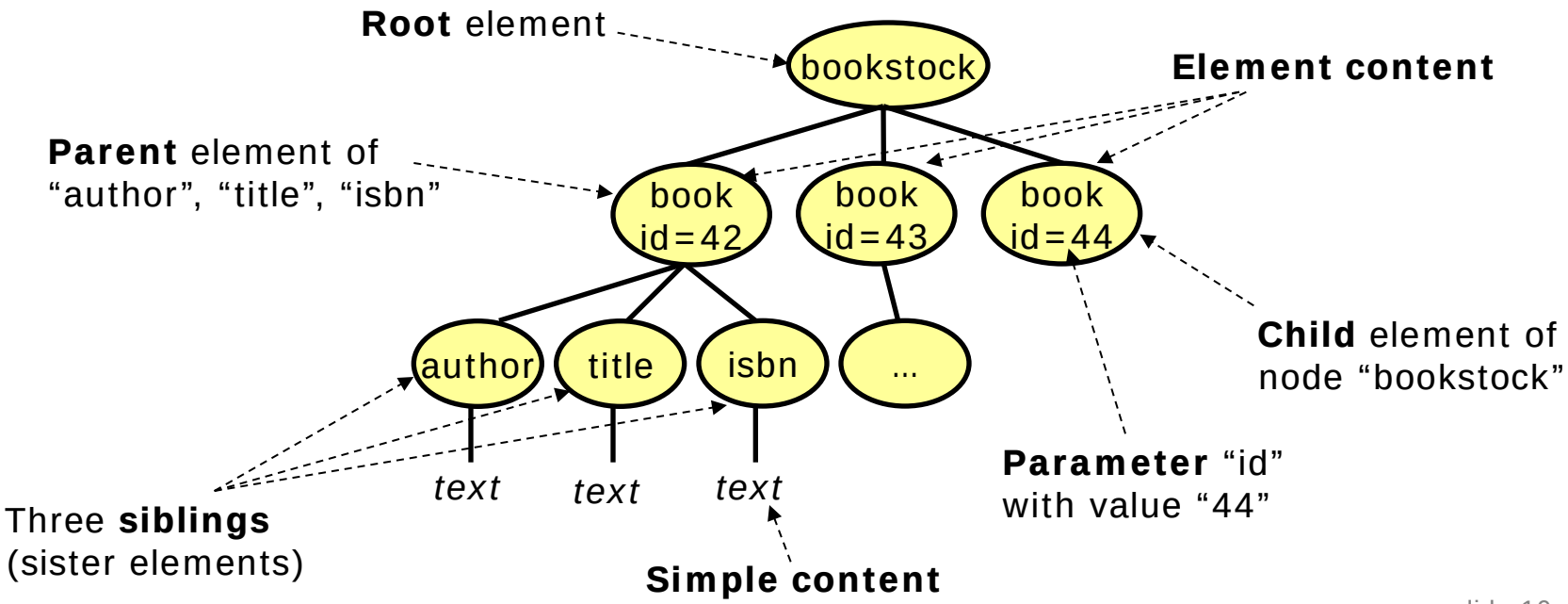
- The Extensible Markup Language (XML) is a general purpose **meta-language** for representation, exchange and processing of data
- Basis for many web related formats



- XML documents are organised in a simple tree structure with the documents' elements as nodes

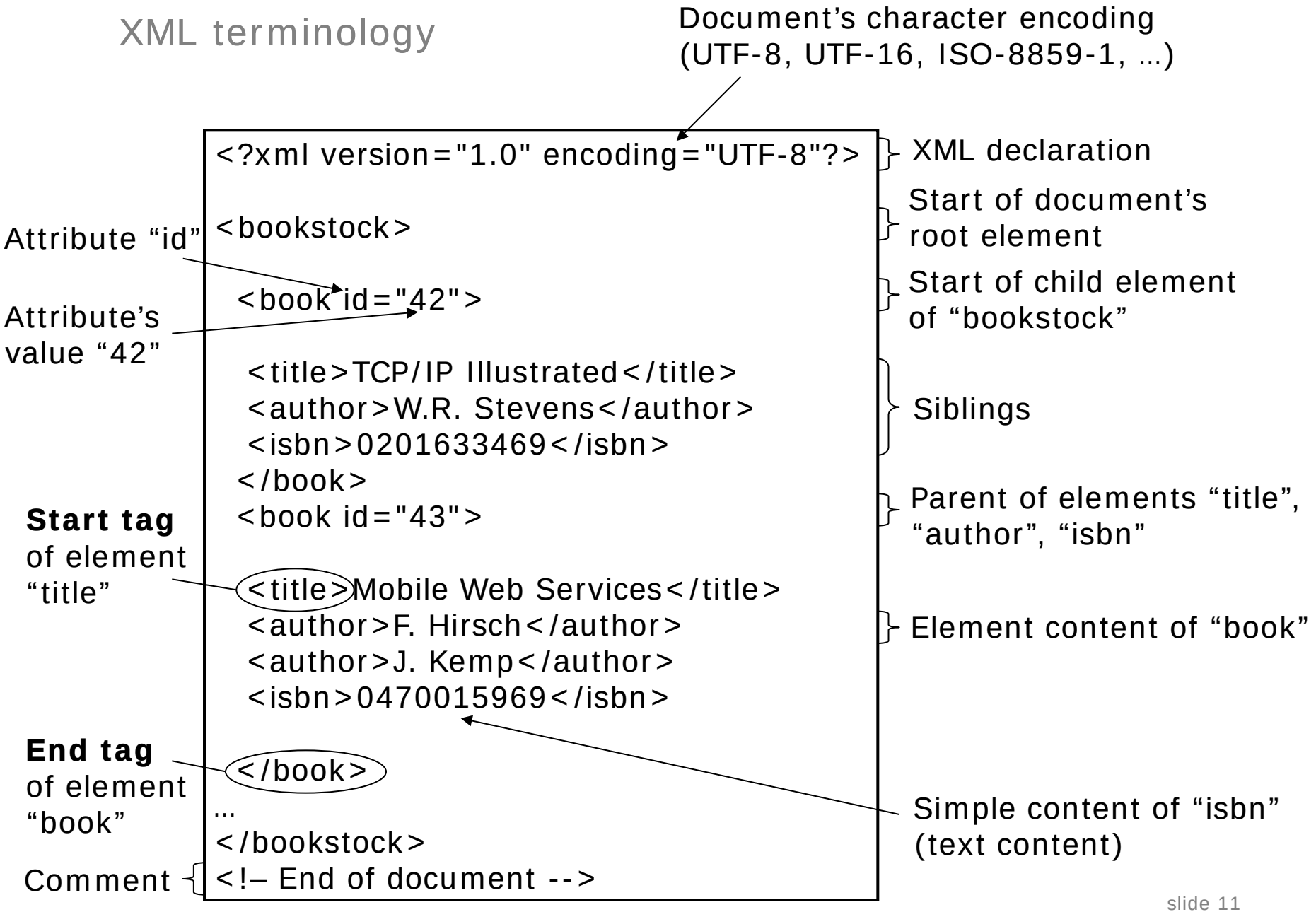
XML terminology

- Every node might have content and parameters
- An element's content is one of
  - Empty content (no value)
  - Simple content (text values)
  - Element content (further "tags")
  - Mixed content (simple and element content)



'text' stands for arbitrary text value

XML terminology



XML syntax

- Main important syntactical rules for XML data are:
  - Non-empty elements need to have a start tag and an end tag (<X>content</X>) - If an element is empty it can directly be delimited (<X/>)
  - Exactly one root element exists
  - All attribute values are quoted (single or double quotes)
  - Nested tags do not overlap (<X><Y></X></Y> is not allowed)
  - XML is case sensitive: <X></x> is incorrect
  - The XML tags, tag content etc. must comply to the given charset (default: UTF-8)
- Syntactically correct XML documents are called **well-formed**

XML namespaces

- Different contributors might use the same element names to refer to different things
- If such elements are mixed in one file **name conflicts** will occur

```
<datafile>
  <author>Charlie Brown</author>
  <title>Book list</title>
  <bookstock>
    <book>
     <title>TCP/IP Illustrated</title>
     <author>W.R. Stevens</author>
     <isbn>0201633469</isbn>
     <price>41,89</price>
    </book>
  </bookstock>
</datafile>
```

If a computer program should find all book titles and scans the XML file for “title” elements it assumes the data file’s title to be a book title

XML namespaces

Imports namespace and binds it to prefix

```
<datafile xmlns:a="http://www.example.com/doc/"
          xmlns:b="http://www.example.com/myNs/" >
  <a:author>Charlie Brown</a:author>
  <a:title>Book list</a:title>
  <b:bookstock>
    <b:book>
      <b:title>TCP/IP Illustrated</b:title>
      <b:author>W.R. Stevens</b:author>
      <b:isbn>0201633469</b:isbn>
      <b:price>41,89</b:price>
    </b:book>
  </b:bookstock>
</datafile>
```

No name conflicts occur because elements are bound to different namespaces

- Name conflicts are solved by **XML namespaces** which are used to qualify elements and attributes by unique identifiers
- These identifiers are web addresses that might point to non existing resources
- A prefix can be associated with one namespace that is attached to all elements that belong to this namespace
- Typically standardised XML dialects define a namespace that is imported by all documents using this dialect

XML validity

- A **valid** XML document is related to and conforms to further information describing its structure and data types used therein
- Two often used type definition languages are:
  1. **Document Type Definition (DTD)**
     - Defines a list of legal elements
     - Referenced from or embedded in related XML document
     - Main shortcoming:
       - Capabilities are limited (especially limited possibilities for type definitions)

```
<!DOCTYPE note [
<!ELEMENT book (title,author,isbn)>
<!ELEMENT title (#PCDATA)>
<!ELEMENT author (#PCDATA)>
<!ELEMENT isbn (#PCDATA)>
]>
```

Element “author” is child element of “book” and of type PCDATA (simple character data)

2. XML Schema

- XML dialect that enables definition of various data types
- Defines two categories of types:
  - Simple types (such as string, integer etc.)
  - Complex types (such as sequences of other types)

Element “book” contains child elements “title”, “author”, “isbn” which are all of type “string”

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" >
<xs:element name="book">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="title" type="xs:string"/>
   <xs:element name="author" type="xs:string"/>
   <xs:element name="isbn" type="xs:string"/>
  </xs:sequence>
 </xs:complexType>
</xs:element>
...
</xs:schema>
```

XHTML

- Besides HTML5 (discussed in chapter 2), one wide-spread application of XML is the Extensible Hypertext Markup Language (XHTML)
- Its language constructs and thus its expressive power were deduced from HTML
- High degree of standardisation and stricter syntax leads to further unification of browser engines
- Use of XML for Hypertext documents makes processing by regular XML tools possible (e.g. for style definition or syntax validation)
- Extensible by further XML languages such as MathML for embedding mathematic expressions into web documents
- XHTML documents may be validated by a DTD
  - E.g. XHTML 1.1:

    http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd

XHTML example

References to associated DTD

Defines default namespace

Empty elements are directly closed

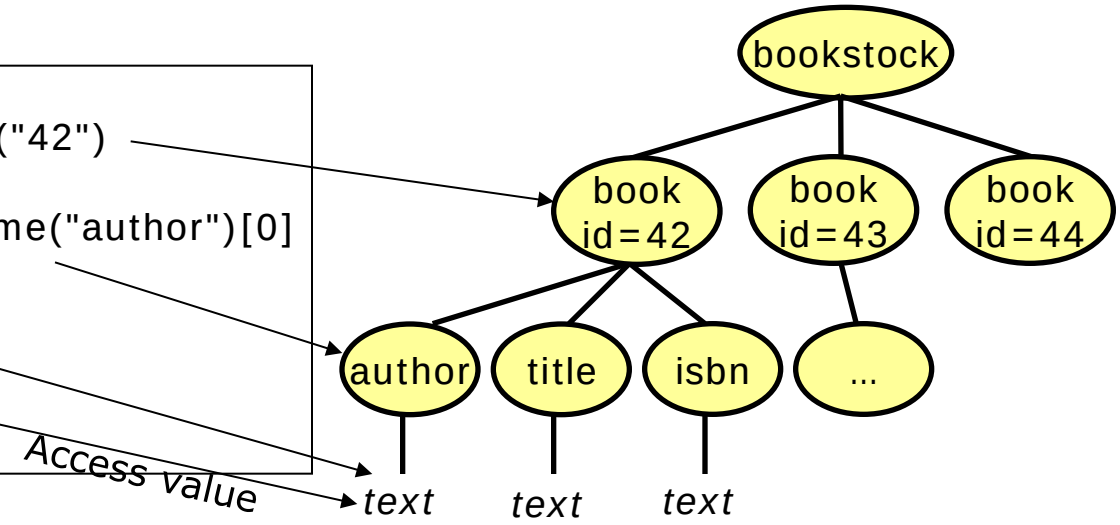
```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
      "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en">
 <head>
  <title>XHTML Example</title>
 </head>
 <body>
  <h3>Available books</h3>
  <br/>
  <table>
  <tr><td>Title</td><td>Author</td></tr>
  <tr><td>TCP/IP Illustrated, Volume 1</td><td>W.R. Stevens</td></tr>
  <tr><td>Mobile Web Services</td><td>F. Hirsch / J. Kemp</td></tr>
  <tr><td>Hacking RSS and Atom</td><td>L.M. Orchard</td></tr>
</table>
 </body>
</html>
```

DOM

- The Document Object Model (DOM) is an application programming interface standardised by the W3C for accessing and manipulating XML documents
- DOM interprets the document in its **logical tree structure** thus making it necessary to load a tree representation of the document into memory
- It renders it possible to traverse through the tree or to directly access elements by an id

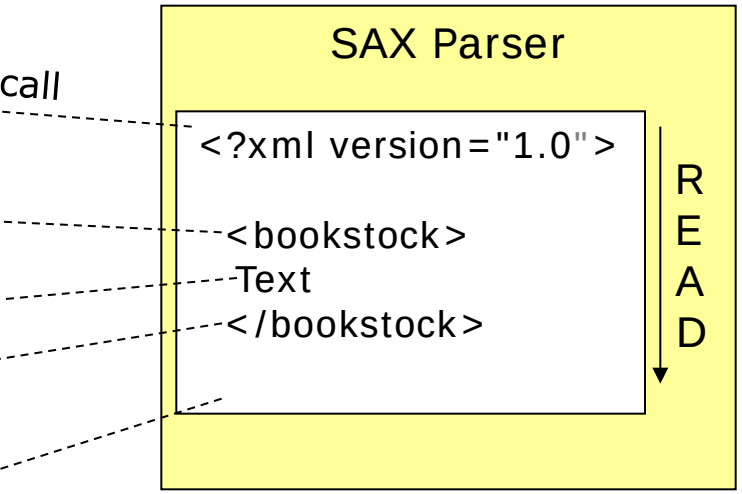
```
var elem = document.getElementById("42")
var child = elem.getElementsByTagName("author")[0]
var secondChild = child.childNode
var value = secondChild.nodeValue
```



getElementsByTagName returns a list of elements that have the given element name

SAX

```
// Application (SAX handler excerpt)
public void startDocument (){
    System.out.println("Start document");
    }
public void startElement (...){
    ...
    }
public void characters (...){
    ...
    }
public void endElement (...){
    ...
    }
public void endDocument (){
    System.out.println("End document");}
```

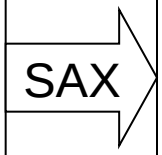


- The Simple API for XML is a **event-based** mechanism for XML processing
- Principle:
  1. A program that e.g. is intended to extract information from an XML file registers event handlers at a SAX parser
  2. The parser reads the XML document sequentially from the beginning to the end
  3. Every time the parser reads an element, attribute, special character etc. it generates an event that is delegated to the associated event handler which then may extract wanted information

SAX example

Input document

```
<?xml version="1.0"
        encoding="UTF-8"?>
<bookstock>
<book id="42">
<title>TCP/IP Illustrated</title>
<author>W.R. Stevens</author>
<isbn>0201633469</isbn>
</book>
<book id="43">
<title>Mobile Web Services</title>
<author>F. Hirsch</author>
<author>J. Kemp</author>
<isbn>0470015969</isbn>
</book>
</bookstock>
```



Event handler output

```
Start document
Start element: bookstock
Characters:    "\n"
Start element: book(Attributes:id=42;)
Characters:    "\n"
Start element: title
Characters:    "TCP/IP Illustrated"
End element: title
Characters:    "\n"
Start element: author
Characters:    "W.R. Stevens"
End element: author
Characters:    "\n"
Start element: isbn
Characters:    "0201633469"
End element: isbn
Characters:    "\n"
End element: book
Characters:    "\n"
. . .
End element: bookstock
End document
```

Selected alternatives to XML

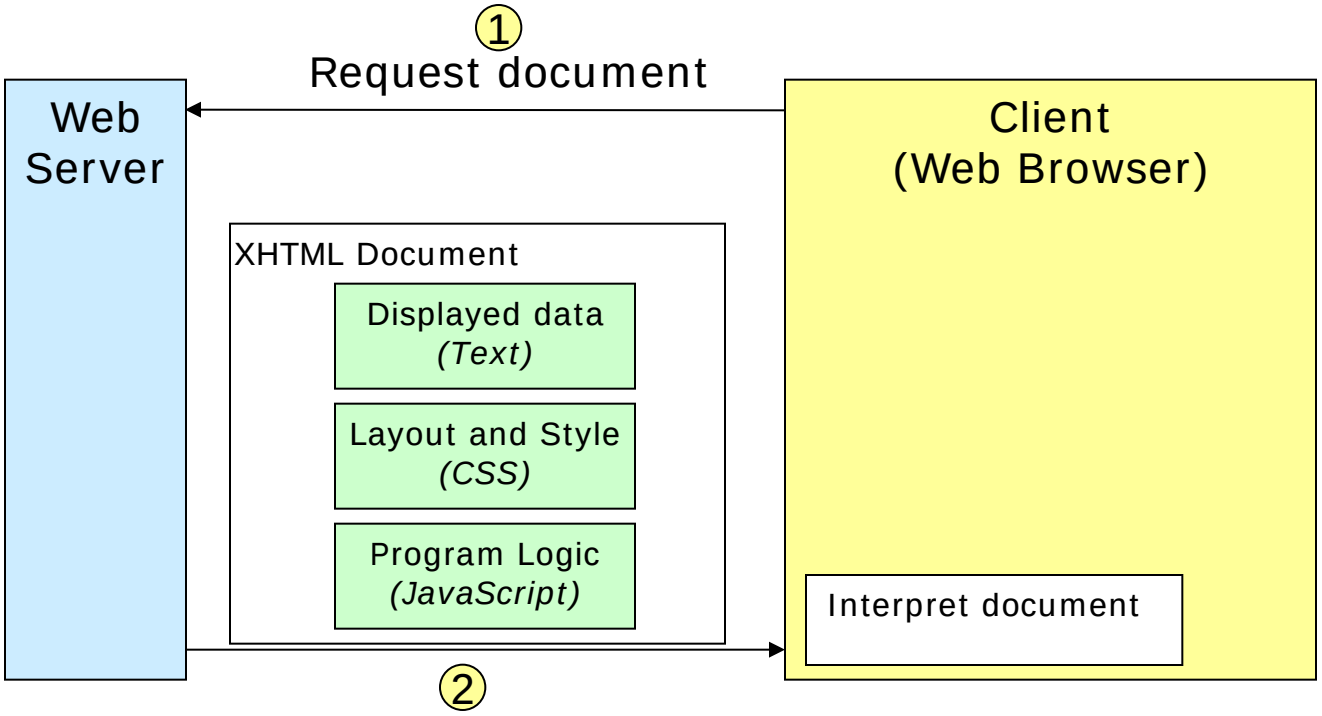
- Though XML is intensively applied, XML documents have a certain overhead and contain redundant information
- Alternatives applied in Web applications include:
  - **JSON** (JavaScript Object Notation, discussed in chapter 3)
  - **YAML** (YAML Ain't Markup Language)
    - Compact text-based and human readable data serilization format
    - Superset of JSON
    - Offers lists, associative arrays and scalars for structuring data
    - Enables cross-references between nodes
    - Hierarchies are realized by indentation

E.g. YAML list in block format

```
lectures:
        - 'Internet and Web Applications'
        - 'Distributed Systems'
        - 'Mobile Communication and Computing'
```

Overview of interaction on the Web

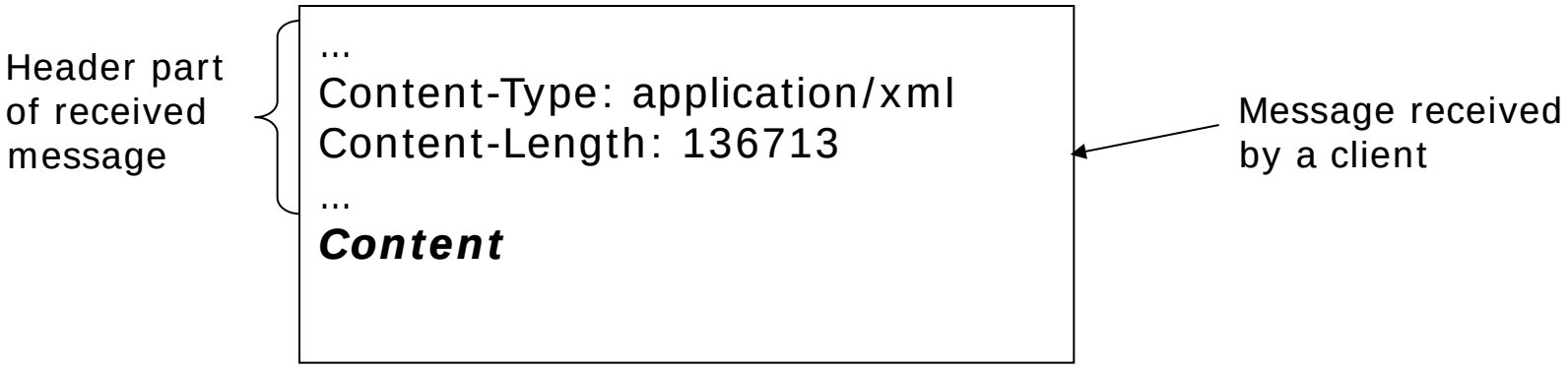
- The Web is based on a classical client-server architecture
- Client fetches **documents** (addressed by an Uniform Resource Identifier) from the server using the Hypertext Transfer Protocol (HTTP)
- Documents may have different types of content (text, images, ...)
- Content of one document may consist of different sections interpreted by the client



Content type

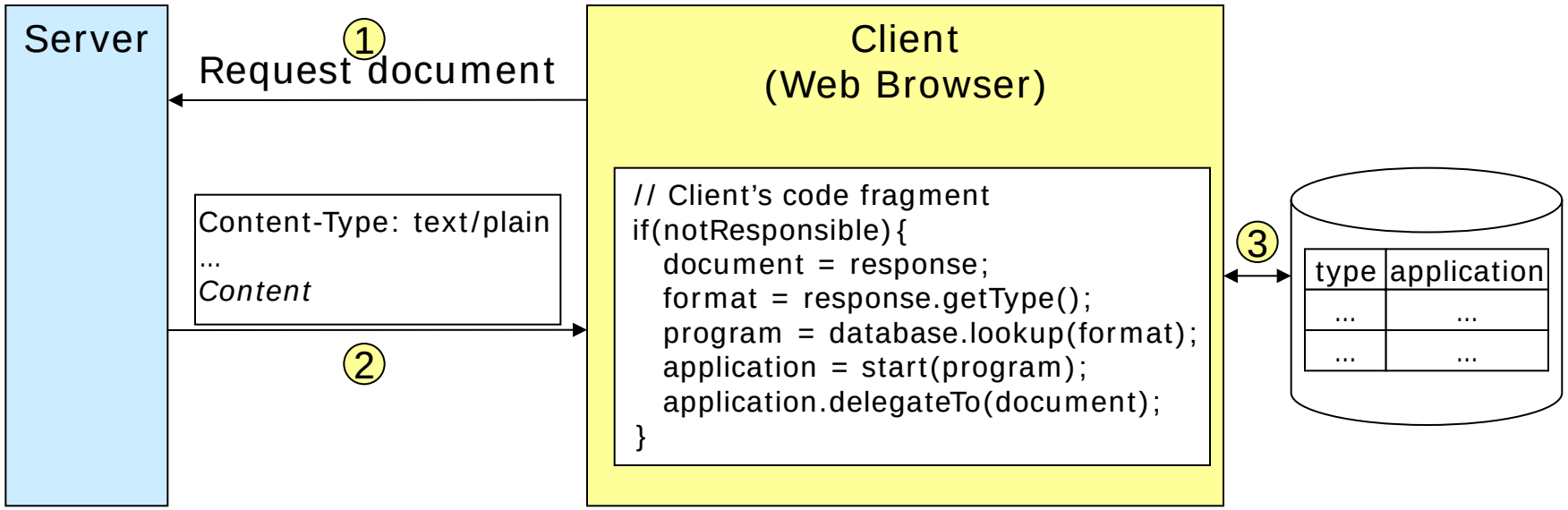
- In order to find an appropriate interpreter program for received content on the client side the content's meta information specifies a content type (sometimes referred to as "media type" or "MIME type")
- Format:

  Content-Type ":" type "/" subtype
- Examples:
  - Content-Type: application/xml *(for general XML content)*
  - Content-Type: video/mpeg *(for MPEG encoded content)*
  - Content-Type: image/jpeg *(for JPEG encoded content)*



Content type

- Client manages data structure that maps known content types to responsible applications
- After the client has received a document it analyses the document's content type information
- If the client is not responsible for the type of content it looks up an appropriate application and initiates handling of the document by this application



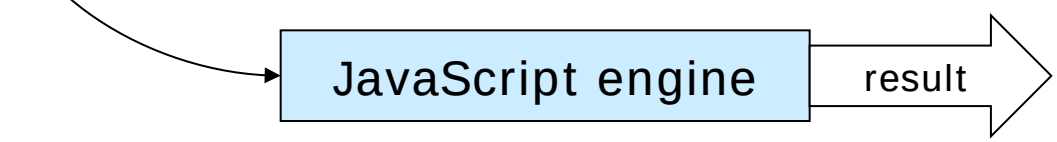
JavaScript

- JavaScript is an **object based** wide-spread script language (standardised under the name 'ECMAScript') often used for client-side execution
- May be embedded into (X)HTML documents or linked from them
- After the web browser has identified code segments it forwards them for execution to the JavaScript engine
- The result of execution might influence the original document's content or force an additional reaction (e.g. open new browser window)

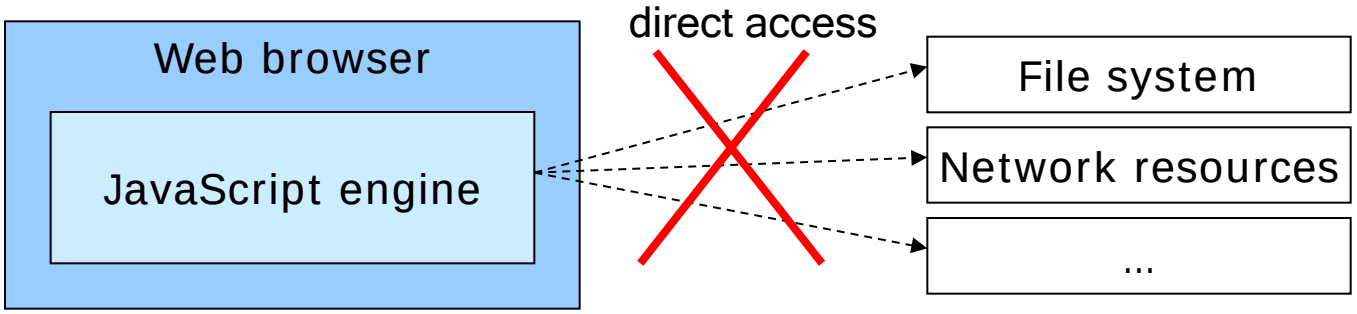
```
<html>
<head>
<script language="JavaScript" type="text/javascript">
function hello() { alert("Hello!"); }
</script>
</head>
<body onLoad="hello()">
This is a simple JavaScript example.
</body>
</html>
```

Definition of function hello

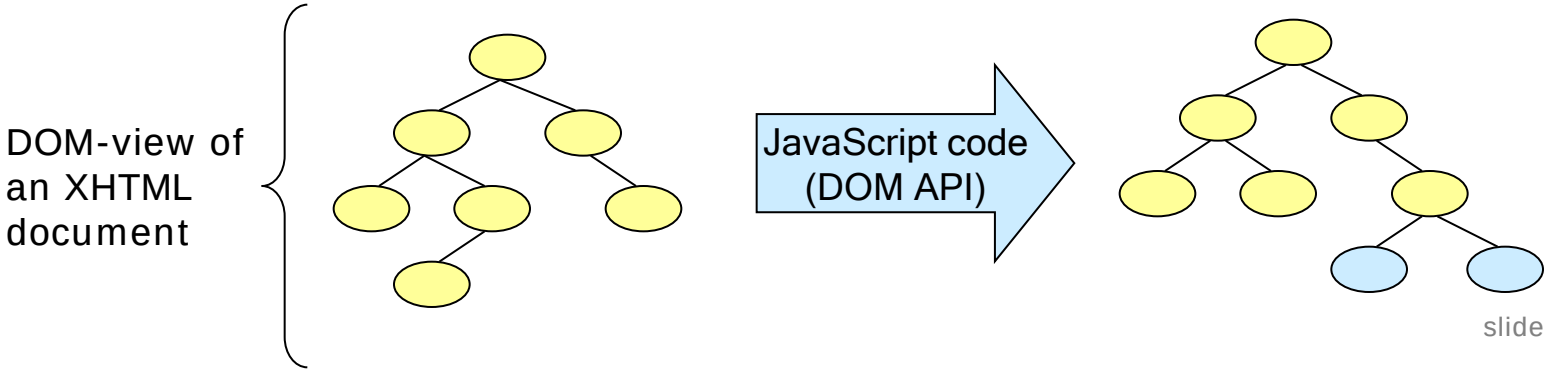
When loading the page the function "hello" is called



- Important security concept is the **sandbox principle** which permits script logic that is running inside the JavaScript engine access only to objects inside the browser per default



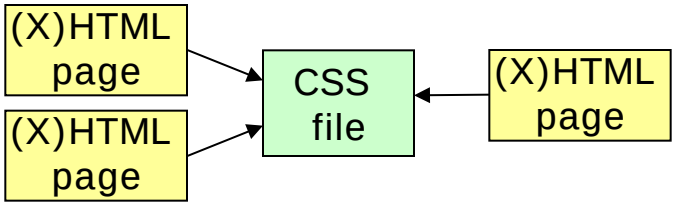
- Java Script is often used in combination with DOM leading to a dynamically changeable structure of an XHTML document



CSS

- Cascading Style Sheet language is a client-side language for defining layout of mark-up language elements
- Simplifies uniform design and its maintenance of web pages
- CSS description may be directly embedded into (X)HTML file or linked from it (e.g. <link rel="stylesheet" href="main.css">)

- If style definitions are given as external file the layout of all associated (X)HTML pages can be changed by editing this single file



- The CSS file contains a number of selectors (e.g. (X)HTML tags or self defined names) whose properties are defined
- These selectors refer to elements of (X)HTML files (e.g. to tag with specific name or attribute)

```
Selector
{
property1: valuesX;
property2: valuesY;
...
}
```

CSS

main.css

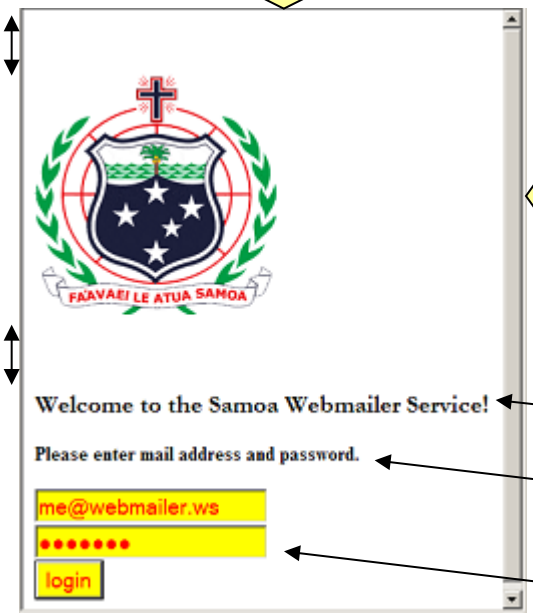
```
input{ font-size: 110%;
       color: red;
       background: #ffff00; }
.top{ padding-top: 20px; }
p{ font-size: 16pt;
   font-family: "Garamond", serif;
   font-weight: bold; }
#fat { font-weight: bold; }
```

index.html

```
<head>…
 <link rel="stylesheet" href="main.css">
</head>
<body>
  <div class="top">
   <img src="/banner.jpg"/>
        </div>
  <p class="top">
  Welcome to the Samoa Webmailer Service!
  </p>
  <div id="fat">
  Please enter mail address and password.</div>
  <form method="post" action="login">
    <input type="text" name="id"/><br/>
    <input type="password" name="password"/>
    <br/><input type="submit" value="login"/>
  </form>
</body>
```

20 pixel

20 pixel



bold; Garamond as font

bold

yellow (#ffff00) background; red font

References

Links at World Wide Web Consortium (W3C):

XML home	http://www.w3.org/XML
XML Schema home	http://www.w3.org/XML/Schema
XHTML 1.0	http://www.w3.org/TR/xhtml1/
XHTML 1.1	http://www.w3.org/TR/xhtml11/
HTML 5	http://www.w3.org/TR/2014/REC-html5-20141028/
DOM home	http://www.w3.org/DOM/

Further Links

YAML-Spezifikation	http://www.yaml.org/spec/1.2/spec.html
SAX home	http://www.saxproject.org/
JavaScript at mozilla.org	https://developer.mozilla.org/en/About_JavaScript

Article about Web 2.0
http://oreilly.com/web2/archive/what-is-web-20.html