WIRTSCHAFTS UNIVERSITÄT WIEN VIENNA UNIVERSITY OF ECONOMICS AND BUSINESS

# HTML 5:
# Concepts, Architectures, Nutshell Examples, Outlook

*Seminar Paper*

*Submitted to*

**Univ.Prof. Dr. Rony G. Flatscher**

Institut für Wirtschaftsinformatik und Gesellschaft,
Wirtschaftsuniversität Wien

*by*

**Stefan Loidolt**

Matriculation Number: h01610844

WIRTSCHAFTS UNIVERSITÄT WIEN VIENNA UNIVERSITY OF ECONOMICS AND BUSINESS

# Table of Contents

WIRTSCHAFTS UNIVERSITÄT WIEN VIENNA UNIVERSITY OF ECONOMICS AND BUSINESS

# List of Figures

# List of Tables

# Table of Abbreviations

**API** | Application Programming Interface

**CERN** | Conseil Européen pour la Recherche Nucléaire

**CSS** | Cascading Style Sheets

**HTML** | Hyper Text Markup Language

**HTTP** | Hypertext Transfer Protocol

**SSE** | Server-Sent Events

**URI** | Universal Resource Identifier

**W3C** | World Wide Web Consortium

**WHATWG** | Web Hypertext Application Technology Working Group

**WWW** | World Wide Web

# 1 Introduction

## 1.1 Objectives of the Seminar paper

This seminar paper deals with the topic of HTML 5. It provides an overview of the origins, the historical development and an outlook for the future. However, the work not only deals with the development of HTML, but also with the specifics and innovations of HTML 5. In addition, short examples of the innovations in HTML 5 are also presented.

## 1.2 Overview of the Work Structure

The first part of the seminar paper introduces HTML. What is HTML? What is it used for? When and by whom was it developed? All these questions are clarified here and a historical overview of the development up to the current status is given.

The second part deals with HTML 5 in detail. For this purpose, the development of HTML 5 is described in more detail at the beginning. Furthermore, the innovations and improvements as well as the HTML 5 Living Standard are explained. The semantics and structure as well as the integration of web APIs and browser compatibility are also dealt with. At the end of this part of the seminar paper, the current relevance of HTML 5 is assessed.

The third part shows and explains some short Nutshell examples, which mainly represent the innovations in HTML 5.

The fourth and final part of this seminar paper provides an outlook on the future of HTML. What further developments are coming? How important will HTML be in the future?

# 2 Introduction to HTML

## 2.1 Introduction to HTML

HTML (Hyper Text Markup Language) is the central markup language of the World Wide Web. It is used to transform text documents into web pages and web applications (Robbins, 2010, S. 1). The main task of HTML is therefore to define the syntax and position of special instructions, which are not displayed by the browser itself, but describe how the content, for example text, images or other media, should be displayed. Basically, the content of a website is divided into different elements or tags. These can then be interpreted and displayed by a web browser. The first version of HTML was written by Tim Berners-Lee in 1993 (Chuck Musciano, 2003, S. 9).

## 2.2 W3C

The World Wide Web Consortium (W3C) was founded in 1994 by Tim Berners-Lee to ensure the long-term growth of the World Wide Web and to exploit its full potential. It is an international community that deals with the development of web standards. The aim is to make the World Wide Web accessible and usable for everyone (W3C, W3C, 2024). The members of the W3C are to create and disseminate new drafts and improvements to the standard through feedback. However, the W3C not only plays a major role in the standardization of HTML but is also responsible for the general standardization of World Wide Web technologies. The consortium therefore also manages numerous standards for other web standards such as HTTP, CSS and XML (Chuck Musciano, 2003, S. 8). The projects on which the W3C is working are called “activities”. Each of these activities has a working group and an interest group. While the working group deals with the development of content, the interest group tries to influence changes and developments. Both of these groups consist of members of the W3C. This process is

monitored by an independent core team of the consortium (Münz & Nefzger, 2004, S. 44 ff).

## 2.3 WHATWG

The Web Hypertext Application Technology Working Group (WHATWG) is an unofficial and open collaboration of web industry companies and interested parties (Hickson, whatwg.org, 2004). It was founded in 2004 by leading web browser companies, including Apple, Opera and Mozilla, as some companies considered the standardization of web technologies by the W3C to be too slow. The main aim of the WHATWG is to develop HTML-based specifications to facilitate and promote the provision of interoperable web applications. The aim is to ensure that web technologies function correctly and error-free on different browsers and platforms. The backward compatibility of existing content is of particular concern to the WHATWG (Pilgrim, 2010, S. 12 ff).

## 2.4 Historical Overview and Development of HTML

### 2.4.1 Tim Berners-Lee and the beginnings of the World Wide Web

The Brit Tim Berners-Lee, also known as the "father of the World Wide Web", worked as a computer scientist at CERN (European Organization for Nuclear Research) in the 1980s and 1990s. There he used a self-written program, which he called "Enquire", as a productivity tool. *Enquire* was a command-line oriented hypertext program without a graphical user interface. According to Münz and Nefzger (2004, S. 42), text files could be edited in this program, which were divided into so-called nodes. For these nodes, which could be data of any kind, there was an associated list in which links to other relevant nodes could be found.

In 1988, Tim Berners-Lee decided to develop *Enquire* into a cross-computer system and submitted a detailed project proposal to CERN in 1989. The

project was given the name "World Wide Web" and Berners-Lee drew up an initial concept. This concept was based on three pillars:

- **HTTP:** Used for communication between web clients and web servers
- **URI:** Used to address files and sources in the WWW
- **HTML:** Markup language for web documents

Berners-Lee made the world's first website written in HTML available at Christmas 1990 via his self-written web server software. This website was accessible at info.cern.ch. The difference to the web browsers that prevailed was that Berners-Lee made the website editable online for authorized visitors so that texts could be written. The other web browsers were merely read-only software. However, the development of the web did not progress on its own. It was only after the international hypertext conference in 1991, at which the project was presented, that the first web browsers were created (Münz & Nefzger, 2004, S. 42 ff).

### 2.4.2 Netscape and Internet Explorer

It was the availability of web browsers that made it possible for people with internet access to access websites. After the first browsers such as "Erwise" or "Viola" were created, Marc Andreessen was able to develop the first usable graphical web browser called "Mosaic". This browser gradually got the WWW rolling. However, the Mosaic project soon became too small for Marc Andreessen, who wanted to develop a browser that would conquer the world. He subsequently became co-founder of Netscape. This web software was able to provide websites with colorful backgrounds, table layouts and multimedia plug-ins. Companies, organizations and authorities entered the market and created their own websites. Münz and Nefzger (2004, S. 43) speak of explosive growth. In 1995 and 1996, the Netscape browser was able to claim a market share of 90% at times. This changed when Microsoft entered the market with Internet Explorer. Microsoft made Internet

Explorer part of the Windows operating system and quickly gained market share but was also heavily criticized. In the end, Microsoft achieved an 80% to 90% share of the web browser market (Münz & Nefzger, 2004, S. 44).

### 2.4.3 Intervention by the W3C

Due to the "browser war" between Netscape and Internet Explorer, new features were introduced without taking other manufacturers into consideration (html, 2024). This made websites with browser switches necessary. These browser switches offer different codes for different browsers in order to prevent differences in the display of websites in different browsers (html, selfhtml - Die Energie des Verstehens, 2024).

These developments were counteracted by the work of the W3C. The technical foundations and standards on the web are developed and published by the W3 consortium. The W3C has become the most powerful factor for further development on the web. This is mainly because the work of the W3C is not directed against the interests of software companies. On the contrary, its members often come from software companies and can contribute to new standards (Münz & Nefzger, 2004, S. 45).

### 2.4.4 HTML to HTML 5

***Table 1:*** *HTML Versions*

| Version | Year of Publication | Innovations |
|---|---|---|
| HTML | 1992 | original version |
| HTML | 1993 | text and image integration |
| HTML + | 1993 | |
| HTML 2.0 | 1995 | formular technology |
| HTML 3.0 | failed | |
| HTML 3.2 | 1997 | tables & applets |

| | | |
|---|---|---|
| HTML 4.0 | 1997 | CSS, scripts & frames |
| HTML 4.01 | 1999 | corrections & updates |
| XHTML 1.0 | 2000 | new formulation of HTML 4.01 with the help of XML |
| XHTML 1.1 | 2001 | division of XHTML into modules |
| XHTML 2.0 | 2006 | has been discontinued |
| HTML 5 | 2014 | new vocabulary, foundation for the standard to this day, based on XHTML |

HTML underwent a number of innovations, improvements and extensions in the early days. For example, the original version was expanded in subsequent years to include text and image integration. HTML 2.0 then also added a form function (Wehrens, 2022). Due to the fact that Netscape was already much further along in its development than the official HTML standard, many were disappointed by the restrictions imposed by HTML 2.0 (Münz & Nefzger, 2004, S. 76). When the development of HTML changed location with the founding of the W3C, HTML 3.0 was worked on but never released, as this version gave way to a more pragmatic approach known as HTML 3.2 (W3C, W3C). In this version, tables and applets were integrated into HTML (Wehrens, 2022). HTML 4.0 was also published in the same year. This version brought stylesheets, scripts, and also frames (Wehrens, 2022). A correction of this version was then published in 1999, after which the members of the W3C decided to discontinue the development of HTML and work on XHTML instead. XHTML 1.0 is a reformulation of HTML 4.01 based on the markup language XML. No new elements were added, only additional rules

for the markup were added (Robbins, 2010, S. 8). While the development work for XHTML 2.0 was underway, some web browser manufacturers made a proposal to the W3C in 2004 to further develop HTML 4. However, this proposal was rejected, resulting in the creation of the WHATWG, which developed its own specifications based on HTML 4.01. Similar to HTML 3.0, XHTML 2.0 was also discontinued as this version was not compatible with HTML or XHTML 1.0 and therefore received little support (Wikibooks, 2014). After the W3C officially ended the XHTML 2.0 project in 2009, all forces will apply HTML 5 in future (Robbins, 2010, S. 8).

## 2.5 Development of the Specifications

Both the W3C and the WHATWG have been working on HTML 5 since 2009. They have different approaches. While the W3C publishes finished standards, the WHATWG adheres to the "Living Standard", which is subject to continuous further developments (WHATWG, HTML, 2024). The reason for the "Living Standard" was the high demand for new functions, which is why the WHATWG decided to further improve HMTL 5 and not to release an "incomplete" HTML 5 version. However, as there is also interest in a snapshot of HTML 5, it was agreed that the W3C would publish this. The two groups work in constant exchange with each other (Hickson, The WHATWG Blog, 2011).

# 3 HTML 5

## 3.1 Introduction to HTML 5

HTML 5 is the latest version of HTML and replaces the previous versions HTML 4.01, XHTML 1.0 and XHTML 1.1. HTML 5 implements new functions that are indispensable for modern web applications. Numerous long-used functions were also standardized and documented with this version. Like the predecessors of HTML 5, this version is cross-platform and therefore only bound to a modern web browser (Pilgrim, 2010).

## 3.2 Living Standard

With the "Living Standard", the WHATWG has embarked on a new path for the further development of HTML. This specification is constantly being further developed and supplemented together with web browser manufacturers. Elements can also be removed again. The WHATWG came to this conclusion because the demand for new elements and functions was still high and further developments were necessary. As a result, they did not want to publish an incomplete HTML 5 and opted for a version that is constantly evolving. In 2011, the WHATWG renamed its specification of HTML 5 to HTML, which underlines this dynamic approach. In return, the W3C publishes snapshots of the current status (Hickson, The WHATWG Blog, 2011).

## 3.3 Basic structure of an HTML document

### 3.3.1 Overview

The basic HTML structure basically consists of three parts:

- **Doctype-Declaration:** The document type declaration is required to give the web browser information about the type of document.
- **Header:** Among other things, information on the title and metadata is provided here.
- **Document-Body:** The document body contains information that ultimately represents the content to be displayed. This can be text, for example, but also graphic references or links.

***Figure 1:*** *Basic Structure of HTML 5-Documents*

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Titel</title>
  </head>
  <body>
    <! -- sichtbarer Inhalt -->
  </body>
</html>
```

***Source:*** *(Münz & Nefzger, 2004, S. 58)*

The figure above shows a basic structure. The first line contains the document type declaration. The remaining content is located between the two tags <html> and </html>. In addition, between the <head> and </head> tags, there is information about the header area. The character encoding used is defined here and the title is defined between the <title> and </title> tags. Below this is the part of the document that is to be displayed in the browser. This is opened by the <body> tag and closed again by </body> (Münz & Nefzger, 2004, S. 59).

### 3.3.2 Doctype-Declaration

The story behind the document type declaration begins with web browsers. During the development of Internet Explorer 5, Microsoft noticed that older websites were no longer rendered responsively. The problem was solved by the browser looking at the document type before rendering the web page, which is at the top of the document before the <html> tag. If there is no document type here, the document is treated as an "old" one. If the new standards are to be used, the correct document type must be specified at the beginning of the document (Pilgrim, 2010, S. 32). In the case of HTML 5, it looks like this: "<!doctype html>". By specifying this, the

browser knows that the document in question is beyond HTML version 4.01 and can render it accordingly (Münz & Nefzger, 2004, S. 60).

### 3.3.3 Header

The first element in an HTML document is the header. According to Münz and Nefzger (2004, S. 69 ff), the following five elements can be placed in this area:

- **Title of HTML-Documents:**
  The title of an HTML document is particularly important as it is displayed both in the title bar of the display window and in browser tabs. It is also used as a title when setting bookmarks and favorites or in the list of websites visited. The title is also very important when searching the web. It is highly relevant for search engines and is listed as a clickable link in search hits. A suitable title is therefore of great importance in terms of search engine optimization alone. Structurally, the title is located between the <title> and </title> tags, which in turn can be found in the header area of the document. It may only appear once, and no other HTML elements are permitted within the title area.

- **Meta-Data:**
  Various useful instructions, such as information about the author or the content of the website, can be included here. Commands that may redirect to other addresses can also be sent. Meta information is specified in the head section of the document and is not limited in

number. It is contained in a so-called "standalone tag". The following illustration shows three such meta specifications:

***Figure 2:*** *Meta-Data*

```
<head>
  <meta charset="utf-8">
  <meta http-equiv="expires" content="0">
  <meta name="author" content="Anna Lyse">
  <!-- ... andere Angaben im Kopfbereich ... -->
</head>
```

***Source:*** *(Münz & Nefzger, 2004, S. 71)*

The first of these three meta specifications refers to the character encoding. It is a special case and was created in HTML 5 to make it easier for authors to specify the character encoding. It is particularly recommended because it makes the transcription of vowel mutation using entities in German languages superfluous. Unfortunately, older browsers often do not recognize this specification, which is why it is recommended to also specify the older version. This is made up as follows:

*<meta http-equiv="content-type" content="text/html; charset=utf-8">*

In addition, this specification is not required if an XHTML 5 document processed by an XML parser is written. In this case, the character encoding specification of the XML declaration applies.

The second meta specification is a so-called pragma directive. In the example above, it indicates that the page content should not be loaded from a cache memory.

In the last meta specification of this example, the name Anna Lyse is assigned to the Author property. Meta specifications with "name=" are mainly used to give search engines properties to fill their search engine database.

- **Relationships and links**

  Logical links can also be specified in the header data of an HTML document. However, such links should not be confused with hyperlinks that point to other documents. The logical relationships involved here point, for example, to the same document in a different language, a style sheet with CSS rules for this document or even a next chapter of an online book. There are also all other links to resources that can be specified by indicating the relationship to the document in question. The links are specified using the <link> tag. Each relationship has its own tag. The type of link is specified with the "rel" attribute and the URL address with the "href" attribute. If the link is displayed in any form in the browser, a name can be added using the "title" attribute. A complete link can therefore look like this in the case of a reference to a website with information about the author:

  *<link rel="author" href="impressum.html" title="Impressum=>*

  The visualization of these links on browser interfaces is not fixed and also depends on the link type. For links to neighboring pages, however, it can make sense to offer corresponding buttons.

- **Stylesheet and JavaScript**

  Style sheets and scripts can be inserted both in the header and in the document body. These areas are enclosed by <style> and </style> as well as <script> and </script>. In both cases, the start tag is intended to inform the browser of the style or script language using the type attribute. If CSS stylesheets and JavaScript are used, the type attribute can be omitted in HTML 5, as these are the default languages.

- **Basis URLs and Hyperlinks**

  So-called base URLs can also be specified in the header area of the HTML document. If hyperlinks are used in the document, these will access the base URLs if they are entered accordingly. This procedure could be outlined as follows.

  In the head section, two base elements can be found. The first specifies the URL base for a following hyperlink, and the second base element determines the name of the browser tab. Furthermore, there could be three hyperlinks in the body of the document that use the URL base, as an example. As a last step, a click on the first hyperlink (<a href="ig">Personal Homepage<a><br>), could open the URL address "http://www.google.com/ig".

### 3.3.4 Document-Body

The last component in the basic structure of an HTML document is the document body. This is marked by the tags <body> and </body>. Here, the entire visible structure of the document is noted. Components of the document body can include sections, text paragraphs, headings, navigation, images, and much more. In the following subchapters, some possible components will be explained in more detail (Münz & Nefzger, 2004, S. 89 ff).

#### *3.3.4.1 Header and Footer*

As in the entire document, there is also a head and foot section within the document body. This is directly visible on the webpage. With HTML 5, these sections can now also be marked. The head section is written between the tags <header> and </header>, and the foot section is written between the tags <footer> and </footer>. In these sections, any other HTML elements can be used but no additional header or footer elements (Münz & Nefzger, 2004, S. 91 ff).

#### *3.3.4.2 Sections*

By adding sections to an HTML document, hierarchies can be clarified. For example, a heading outside the section would be ranked higher hierarchically than headings within the section. In a way, a subdocument is created. This area is marked with the start tag <section> and the end tag </section>. Nesting multiple sections is possible (Münz & Nefzger, 2004, S. 90 ff).

#### *3.3.4.3 Navigation Bar and Sidebar*

Most common websites also feature navigation bars and sidebars. These are columns that are usually located on the left and right edges. Navigation bars are created with the HTML tags <nav> and </nav> and typically contain hyperlinks that refer to other pages on the website. However, other HTML elements or text can also be used within the navigation bar.

Sidebars are used for additional information or advertisements. Any various HTML elements can also be used, and the tags <aside> and </aside> define the area (Münz & Nefzger, 2004, S. 94).

#### *3.3.4.4 Figures*

Figures are marked by the tags <figure> and </figure> and can have a caption using <figcaption> and </figcaption>. The caption must be either at the beginning or at the end of the figure section. Otherwise, the section can contain any other HTML elements besides text. Commonly, these are contents that are set apart from the main text, such as graphics, multimedia elements, labeled tables, or videos (Münz & Nefzger, 2004, S. 109 ff).

## 3.4 HTML 5 Elements

With HTML 5, several new semantic elements were created. In the following, a few significant elements defined by the HTML 5 specification will be explained in more detail. Some of these elements were already covered in

Chapter 3.3, so they are listed here for completeness, but the explanation refers to the above.

### 3.4.1 <section>

The explanation of the section element in HTML 5 can be found in Chapter 3.3.4.2.

### 3.4.2 <nav>

The explanation of the section element in HTML 5 can be found in Chapter 3.3.4.3.

### 3.4.3 <article>

The article tag represents independent, self-contained content within a document. The content of the article element makes sense independently of the rest of the webpage content and can therefore be reused. It is possible to nest article elements. In this case, the inner article elements are related to the outer article element. An example of this is a user's comment on a blog post. Here, the blog post represents the outer article element, and the comment represents the inner article element. Other examples where the article element might be used include newspaper articles, news stories, forum posts, magazine articles, interactive widgets, or numerous other independent contents (WHATWG, HTML, 2024). The start and end tags are created as usual with <article> and </article>. The article element has purely semantic functions. It has no visual effects on the browser display (Münz & Nefzger, 2004, S. 98).

### 3.4.4 <details>

The <details> tag is often used to create an interactive widget with additional information that can be toggled on and off as needed. This functionality is commonly used for FAQs, definitions, or additional explanations. In conjunction with the <details> tag, the <summary> tag is often used, which is covered in Chapter 3.4.5 (W3Schools, W3Schools).

### 3.4.5 <summary>

The <summary> tag defines a visible heading for the <details> element. The heading can be clicked to reveal or hide the details (W3Schools, W3Schools).

### 3.4.6 <aside>

The explanation of the navigation element in HTML 5 can be found in Chapter 3.3.4.3.

### 3.4.7 <hgroup>

The <hgroup> tag in HTML is used to group together related headings, such as main titles and subtitles, into a single unit. This helps to clarify the semantic structure of the page by making it clear that the grouped headings collectively represent a cohesive piece of information (WHATWG, HTML, 2024).

### 3.4.8 <main>

The <main> tag marks the main content of a document, which should be unique to that page. It should not include repeated elements such as sidebars, navigation, copyright information, logos, or search forms. There should be only one <main> element per document, and it should not be placed within <article>, <aside>, <footer>, <header>, or <nav> elements (W3Schools, W3Schools).

### 3.4.9 <header>

The explanation of the navigation element in HTML 5 can be found in Chapter 3.3.4.1.

### 3.4.10 <footer>

The explanation of the navigation element in HTML 5 can be found in Chapter 3.3.4.1.

### 3.4.11 <time>

The <time> element represents its content along with a machine-readable version of it in the datetime attribute. The content is limited to various types of dates, times, time zones, and durations. The <time> element is particularly useful for applications that process and display data, such as calendars, event schedulers, news articles, and blog posts (WHATWG, HTML, 2024).

### 3.4.12 <mark>

The <mark> element highlights or emphasizes a section of text that is relevant for reference purposes or has been highlighted in another context. It can be used in quotations or text blocks to indicate a portion of text that was not originally present. Another example of using this element is a section of a document that has been highlighted due to its high relevance (WHATWG, HTML, 2024).

### 3.4.13 Graphical HTML Elements

With HTML 5, two graphical elements, <canvas> and <svg>, were introduced. They allow for the display and creation of graphics directly in the web browser. While <canvas> is more suitable for dynamic graphics that need frequent updates, <svg> offers the advantage that graphics maintain high quality even when the original size changes.

#### *3.4.13.1 <canvas>*

The <canvas> element allows the creation and display of pixel-based graphics on a webpage in real-time. However, the HTML element <canvas> only represents an area for the graphics. The graphics are created using JavaScript code. This element is used, for example, in games, animations, or graphical effects that need frequent updates. The element is supported by all major browsers (w3schools, w3schools).

#### *3.4.13.2 <svg>*

The HTML 5 <svg> element is used to embed graphics in HTML. These are vector-based graphics. This means that the graphics do not suffer any loss of quality when zoomed or resized. As a result, the <svg> element is particularly suitable for logos, diagrams, or other graphics that should maintain unchanged quality in different sizes. SVG can be embedded directly in the HTML document and can also be edited via HTML code, as it defines graphics in XML format. <svg> elements can also be made interactive using JavaScript or CSS (w3schools, w3schools).

### 3.4.14 Multimedia Elemente

Multimedia elements on the web exist in a variety of formats. They can include music, sound, images, animations, videos, and even movies. Previously, displaying such content required an external plug-in. However, HTML 5 has made it possible to embed multimedia content directly into web pages without the need for external plug-ins (w3schools, w3schools).

#### *3.4.14.1 <audio>*

The <audio> element allows for the embedding of audio files without the need for external plug-ins. The embedded file can be customized and controlled through several attributes:

- **„src":** This attribute specifies the source of the file.
- **„controls":** This attribute enables controls such as pause, volume, play, etc.
- **„autoplay":** This attribute allows playback to start immediately when the page has loaded.
- **„loop":** This attribute causes the file to restart immediately after it has ended.
- **„muted":** The audio file is muted by default; sound must be activated by the user.

The element supports various audio formats such as WAV, Ogg, and MP3 (SelfHtml, SelfHtml, 2024).

#### *3.4.14.2 <video>*

The <video> element allows for embedding video files such as videos or movies without the need for external plug-ins. The embedded file can be customized and controlled through several attributes:

- **„src":** This attribute specifies the source of the file.
- **„controls":** This attribute enables controls such as full screen, pause, volume, playback, etc.
- **„autoplay":** This attribute allows playback to start immediately when the page has loaded.
- **„loop":** This attribute causes the file to restart immediately after it has ended.
- **„muted":** The video file is muted by default; sound must be activated by the user.
- **„preload":** This attribute determines whether the video file should be preloaded or not.

Video formats such as MP4, Ogg, and WebM are supported (SelfHtml, Self-Html, 2023).

## 3.5 Web APIs

An API is an interface that enables data exchange between software and program components. The embedded modules simplify programming and provide a variety of functions for creating interactive web applications (GeeksforGeeks, Geeksforgeeks, 2022). The following will cover five important Web APIs.

### 3.5.1 Geolocation API

The Geolocation API allows for determining the client's location, which is particularly useful for providing users with personalized responses to search queries, targeted advertising, and regional information. In addition to location data, the accuracy of coordinates, speed, and direction of movement can also be captured (SelfHtml, SelfHtml, 2022).

### 3.5.2 Drag-and-Drop API

The Drag and Drop API in HTML 5 allows for dragging elements on a webpage using the mouse and dropping them elsewhere. This enables users, for example, to drag files into an upload area or rearrange elements within the page (W3Schools, W3Schools).

### 3.5.3 Web Storage API

The Web Storage API in HTML 5 is used for storing data in the user's browser. It allows for storing data either locally (localStorage) or temporarily for the duration of the session (sessionStorage), which is useful for managing user sessions, settings, or other information that needs to persist between page visits (SelfHtml, SelfHtml, 2024).

### 3.5.4 Web Worker API

The Web Worker API in HTML 5 is used to execute JavaScript code in the background without affecting the webpage. While the web worker operates in the background, interaction with the webpage can continue. The Web Worker API allows for complex calculations or tasks to be executed in parallel with the main application, improving the responsiveness and performance of the web application (W3Schools, W3Schools).

### 3.5.5 SSE API

Server-Sent Events (SSE) allow a webpage to automatically receive updates from a server. Previously, this was also possible, but the webpage had to regularly inquire whether updates were available. With Server-Sent Events,

updates come automatically and without user intervention (GeeksforGeeks, geeksforgeeks, 2024).

## 3.6 Browser Compatibility

HTML 5 is the current standard for structuring and presenting web content and was first published by the W3C in 2014. A webpage rarely consists solely of HTML code, and for dynamic styling, JavaScript, CSS, and various databases are also required. However, HTML 5 forms the foundation upon which webpages are built, and therefore, it is crucial for browsers to support this standard to correctly display web content.

Fortunately, practically all major browsers support HTML 5. Among the most well-known are Google Chrome, Mozilla Firefox, and Microsoft Edge. But even less widespread browsers like Opera, Chromium, and Tor adopt the new standard. Apple's Safari is also on board and supports HTML 5. This broad support ensures consistent display of web content across different platforms and devices.

Ultimately, the choice of browser depends on personal preferences and requirements. However, regardless of which browser you use, you can be sure that almost all websites will be displayed correctly, as HTML 5 has now become the standard and is extensively supported by most modern browsers (ComputerBild, 2019).

# 4 Nutshell Examples

After delving into HTML 5 in detail in the last chapter, where the structure of an HTML document and new features were explained, this chapter will present and explain some short examples with individual significant elements of HTML 5. This will illustrate the implementation of the features.

## 4.1 Navigation Bar

In the first nutshell example, we will demonstrate how the implementation of a navigation bar can look like. First, the HTML 5 document, or a portion thereof, will be shown, followed by an explanation of the code step by step.

***Figure 3:*** *Nutshell Example <nav>*

```
1  <html>
2   <body>
3
4     <nav>
5     <a href="#home">Startseite</a>
6     <a href="#about">Über uns</a>
7     <a href="#services">Dienstleistungen</a>
8     <a href="#contact">Kontakt</a>
9     </nav>
10
11    <section id="home">
12    <h1>Startseite</h1>
13    <p>Willkommen auf unserer Startseite!</p>
14    </section>
15
16    <section id="about">
17    <h1>Über uns</h1>
18    <p>Hier erfahren Sie mehr über uns.</p>
19    </section>
20
21    <section id="services">
22    <h1>Dienstleistungen</h1>
23    <p>Informationen zu unseren Dienstleistungen.</p>
24    </section>
25
26    <section id="contact">
27    <h1>Kontakt</h1>
28    <p>Kontaktieren Sie uns hier.</p>
29    </section>
30
31   </body>
32  </html>
```

In the code above, we see a short example of the <nav> element. It is limited in structure to the document body, which is opened by the <body> tag in line 2. In line 4, we then open our <nav> element to start creating the navigation bar. For this purpose, links to the various sections of the website are created in lines 5 to 8. The <a> tag is used to create a link. Using "href" (Hypertext Reference), the destination of this link is specified, and the "#" symbol indicates that it refers to a section on the same webpage. The labels "Home", "About Us", "Services", and "Contact" are

what the user ultimately sees and can click on. If clicked, they will land on the linked section of the webpage. Line 9 closes the <nav> element again.

The links are not yet functional because there are no sections with the corresponding IDs. These sections are created starting from line 11. For example, a section is created for the first item in the navigation bar (Home). This section is given the ID "home". This ID now corresponds to the "href" attribute of the hyperlink in the navigation area. So, when you click on "Home" in the navigation bar, you will be directed to the "home" section. In lines 12 and 13, a heading at the highest hierarchy level is created, and a paragraph is created using the <p> tag. Line 14 then closes the section, and the first link in the navigation bar is fully functional. This process is repeated for the other navigation items in the subsequent code. Through these steps, a clear and functional structure for the website has been established.

## 4.2 Canvas

In this nutshell example, we will demonstrate how to implement pixel-based graphics using the <canvas> element. First, the HTML 5 document, or a portion thereof, will be shown, followed by an explanation of the code step by step.

**Figure 4:** Nutshell Example <canvas>

```
1  <html>
2    <body>
3      <canvas id="SeminarCanvas" width="400" height="400" style="border:1px solid #000;"></canvas>
4      <script>
5        var canvas = document.getElementById(`SeminarCanvas`);
6        var context = canvas.getContext(`2d`);
7
8        context.fillStyle = 'green';
9        context.fillRect(50, 50, 150, 100);
10
11       context.beginPath();
12       context.moveTo(50, 300);
13       context.lineTo(350, 300);
14       context.strokeStyle = 'green';
15       context.lineWidth = 5;
16       context.stroke();
17
18       context.font = '20px Arial';
19       context.fillStyle = 'black';
20       context.fillText('Hallo BIS-Seminar!', 100, 350);
21     </script>
22   </body>
23 </html>
```

In this example, we see the usage of the <canvas> element from HTML 5. It is used within the document body as shown in line 2. The <canvas> element in line 3 represents the graphic area where images and graphics can be created. An id attribute is used to provide a unique identifier. In our example, this is "SeminarCanvas". Furthermore, the width and height for the graphic area are specified. In the "style" attribute, a black border around the graphic area is created using CSS style.

The creation of a graphic is facilitated between the <script> and </script> tags. JavaScript code is used between these tags. In line 5, the variable canvas is assigned the document "SeminarCanvas". The getcontext method is used under the variable context, which returns a 2D drawing object. This method is used for drawing graphics on the graphic area.

In the next step (from line 8), various graphics are created. The first graphic element created is a green filled rectangle. In line 8, the color is specified, and in line 9, the parameters for the rectangle are passed according to the following format: (x, y, width, height). Here, the parameters x and y specify the position of the top-left corner, and "width" and "height" specify the width and height of the rectangle, respectively.

In the second code block within the <script> area, a green line is drawn. In line 11, a path is initiated, which can be a series of points, lines, or curves. Next, the pen is moved to a position without drawing a line using "moveto(x, y)". Then, a line is drawn from the current position to the new coordinates using "lineto(x, y)". In this example, a line is drawn from coordinates (50, 300) to coordinates (350, 300). The color is set in line 14, and the width of the stroke is set in line 15. Finally, the line is actually drawn onto the graphic area using "stroke()".

In the last code block within the <script> area, text is drawn. In line 18, the font and size of the font are set. Then, the font color is set to black. In

line 20, the text and the coordinates where the text should appear are specified.

## 4.3 SVG

In this nutshell example, it will be demonstrated how to implement graphics using the <svg> element. First, the HTML 5 document, or a portion thereof, will be shown, followed by an explanation of the code step by step.

***Figure 5:*** *Nutshell Example <svg>*

```
1  <html>
2   <body>
3
4     <svg width="400" height="400" style="border:1px solid #000;">
5
6        <rect x="50" y="50" width="150" height="100" fill="green" />
7
8        <line x1="50" y1="300" x2="350" y2="300" stroke="green" stroke-width="5" />
9
10       <text x="100" y="350" font-family="Arial" font-size="20" fill="black">Hallo BIS-Seminar!</text>
11
12    </svg>
13   </body>
14 </html>
```

In this example, we see how we can create graphics using the <svg> element. The SVG area is opened in the document body. In the start tag, the size of the graphic area is specified, and a black border is added using CSS. As you may have noticed, exactly the same graphic area and graphic elements as in the example using the <canvas> tag are created. This allows us to compare the clarity of the code.

In line 6, a green rectangle is created with its top-left corner positioned at x=50 and y=50. Then, the width, height, and fill color are specified.

In line 8, the start and end points of a line are specified. The color and width of the line are also defined.

In line 10, the text "Hello BIS-Seminar!" is created. The position is specified using coordinates, and the font, font size, and font color are also provided.

Subsequently, the SVG area is closed.

## 4.4 Video File

This example demonstrates how the <video> element can be utilized.

***Figure 6:*** *Nutshell Example: <video>*

```
1  <html>
2   <body>
3
4     <video width="320" height="240" controls>
5       <source src="video.mp4" type="video/mp4">
6       Ihr Browser unterstützt das Video-Tag nicht.
7     </video>
8
9   </body>
10 </html>
```

In the start tag of <video>, the width and height of the video display element are specified. Additionally, we see the "controls" attribute, which ensures that control elements such as play, pause, and volume control are displayed.

In line 5, the source of the video is specified. In this case, the path to the video file is "video.mp4" and the type of the video file is "video/mp4".

In line 6, we see text that will be displayed if the web browser cannot display the <video> element. Then, the area is closed again.

## 4.5 Geolocation API

In this last nutshell example, we will illustrate how the implementation of a Geolocation API could look like. It determines and displays the current location of a user.

***Figure 7:*** *Nutshell Example Geolocation API*

```
1  <html>
2    <body>
3
4      <button onclick="getLocation()">Standort ermitteln</button>
5      <p id="location"></p>
6
7      <script>
8        function getLocation() {
9          if (navigator.geolocation) {
10           navigator.geolocation.getCurrentPosition(showPosition, showError);
11         } else {
12           document.getElementById("location").innerText = "Geolocation wird von diesem Browser nicht unterstützt.";
13         }
14       }
15
16       function showPosition(position) {
17         var lat = position.coords.latitude;
18         var lon = position.coords.longitude;
19         document.getElementById("location").innerText = "Breitengrad: " + lat + "\nLängengrad: " + lon;
20       }
21
22     </script>
23   </body>
24 </html>
```

In the HTML section of this code, a button is created in line 4 with the label "Get Location", which calls the JavaScript function "getLocation()". In the next line, a paragraph element is created and assigned the ID "location". This element is used to display the retrieved location information.

From line 7 to line 22, JavaScript is used between the <script> tags. The first JavaScript function, "getLocation()", checks in line 9 whether the Geolocation API is supported by the browser. If so, the "getCurrentPosition()" method is called. This method requests the user's location information. The method contains two callback functions: "showPosition" and "showError". If the API is not supported by the browser, the message from line 12 is written into the paragraph element.

If the location information is retrieved correctly, the function "showPosition(position)" is called. This function extracts the latitude and longitude from the "position" object and sets these coordinates into the

paragraph element with the ID "location". As a result, the user can see the location information.

If an error occurs and the location information cannot be retrieved, the "showError" function is called. This function was not included in the example, but depending on the reason for the error, it could provide various feedback messages to the user.

# 5 Summary and Future Outlook of HTML 5

In summary, HTML 5 has gained immense significance and relevance in web development, which should not be underestimated. As the latest version of Hypertext Markup Language (HTML), HTML 5 offers a wealth of features and possibilities that are fundamentally changing the web landscape.

A central aspect of HTML 5 is the seamless integration of multimedia content such as audio and video directly into web pages, without relying on external plugins. This significantly enhances the user experience and ensures better compatibility with various devices and platforms.

Furthermore, the new semantic elements like <header>, <footer>, <article>, and <section> simplify the structuring of web content, which is beneficial for both developers and search engines. Improvements in forms, including new input types and validation options, also contribute to user-friendliness.

HTML 5 also allows for the creation of dynamic graphics and animations directly in the browser using Canvas and SVG, without relying on external plugins. Additionally, support for offline functionality through Web Storage and the Application Cache contributes to the reliability and performance of web applications.

In today's digital landscape, HTML 5 is crucial. It's essential for developing mobile web applications as it enables a consistent user experience across different devices and operating systems. Additionally, HTML 5 supports the

creation of interactive web applications such as online games, multimedia presentations, and real-time communication applications.

HTML 5's role as a standard for web development makes it future-proof and is expected to continue playing a central role as it is constantly evolving to meet changing requirements and technologies. Additionally, HTML 5 contributes to making web content accessible, thus reaching a wider range of users regardless of their abilities or limitations. Overall, HTML 5 is an indispensable part of modern web development and plays a crucial role in creating innovative, user-friendly, and accessible web content.

# List of Sources


Chuck Musciano, B. K. (2003). HTML & XHTML: das umfassende Referenzwerk. Köln: O´Reilly.

ComputerBild. (23. 05 2019). *ComputerBild*. Abgerufen am 05 2024 von https://www.computerbild.de/artikel/cb-Tipps-Internet-Welche-Browser-unterstuetzen-HTML5-Ein-kleiner-ueberblick-31519421.html#:~:text=Eigentlich%20jeder%20bekannte%20Browser%2C%20der,ebenfalls%20den%20neuen%20Standard%20nutzen.

GeeksforGeeks. (22. 11 2022). *Geeksforgeeks*. Abgerufen am 05 2024 von https://www.geeksforgeeks.org/why-apis-are-consumed-in-html-5/

GeeksforGeeks. (11. 03 2024). *geeksforgeeks*. Abgerufen am 05 2024 von https://www.geeksforgeeks.org/what-is-server-sent-events-in-html5/

Hickson, I. (04. 06 2004). *whatwg.org*. Abgerufen am 05 2024 von https://whatwg.org/news/start

Hickson, I. (19. 01 2011). *The WHATWG Blog*. Abgerufen am 05 2024 von https://blog.whatwg.org/html-is-the-new-html5

html, s. (30. 01 2024). *selfhtml - Die Energie des Verstehens*. Abgerufen am 05 2024 von https://wiki.selfhtml.org/wiki/Browserweiche

html, s. (12. 03 2024). *selfhtml - Die Energie des Verstehens*. Abgerufen am 05 2024 von https://wiki.selfhtml.org/wiki/Grundlagen/Was_ist_das_Internet%3F

Münz, S., & Nefzger, W. (2004). *HTML § Web-Publishing Handbuch.* Poing: Franzis.

Pilgrim, M. (2010). *HTML5: up and running.* Sebastopol: O`Reilly.

Robbins, J. N. (2010). *HTML & XHTML - kurz & gut.* Köln: O´Reilly.

SelfHtml. (13. 06 2022). *SelfHtml*. Abgerufen am 05 2024 von https://wiki.selfhtml.org/wiki/JavaScript/Geolocation

SelfHtml. (27. 11 2023). *SelfHtml*. Abgerufen am 05 2024 von https://wiki.selfhtml.org/wiki/HTML/Elemente/video

SelfHtml. (25. 03 2024). *SelfHtml*. Abgerufen am 05 2024 von https://wiki.selfhtml.org/wiki/JavaScript/Web_Storage

SelfHtml. (19. 04 2024). *SelfHtml*. Abgerufen am 05 2024 von https://wiki.selfhtml.org/wiki/HTML/Elemente/audio

W3C. (2024). *W3C*. Abgerufen am 5 2024 von About us: https://www.w3.org/about/

W3C. (kein Datum). *W3C*. Abgerufen am 05 2024 von https://www.w3.org/TR/2012/CR-html5-20121217/introduction.html#history-0

w3schools. (kein Datum). *w3schools*. Abgerufen am 05 2024 von https://www.w3schools.com/html/html5_canvas.asp

w3schools. (kein Datum). *w3schools*. Abgerufen am 05 2024 von https://www.w3schools.com/html/html5_svg.asp

w3schools. (kein Datum). *w3schools*. Abgerufen am 05 2024 von https://www.w3schools.com/html/html_media.asp

W3Schools. (kein Datum). *W3Schools*. Abgerufen am 05 2024 von https://www.w3schools.com/tags/tag_details.asp

W3Schools. (kein Datum). *W3Schools*. Abgerufen am 05 2024 von https://www.w3schools.com/tags/tag_main.asp

W3Schools. (kein Datum). *W3Schools*. Abgerufen am 05 2024 von https://www.w3schools.com/tags/tag_summary.asp

W3Schools. (kein Datum). *W3Schools*. Abgerufen am 05 2024 von https://www.w3schools.com/html/html5_draganddrop.asp

W3Schools. (kein Datum). *W3Schools*. Abgerufen am 05 2024 von https://www.w3schools.com/html/html5_webworkers.asp

Wehrens, M. (26. 05 2022). *Computer Bild*. Abgerufen am 05 2024 von https://www.computerbild.de/artikel/cb-Tipps-Internet-HTML-alle-Versionen-Uebersicht-31539199.html

WHATWG. (27. 05 2024). *HTML*. Abgerufen am 05 2024 von https://html.spec.whatwg.org/multipage/introduction.html

WHATWG. (31. 05 2024). *HTML*. Abgerufen am 05 2024 von https://html.spec.whatwg.org/multipage/sections.html#the-article-element

WHATWG. (31. 05 2024). *HTML*. Abgerufen am 05 2024 von https://html.spec.whatwg.org/#the-hgroup-element

WHATWG. (31. 05 2024). *HTML*. Abgerufen am 05 2024 von https://html.spec.whatwg.org/#the-time-element

WHATWG. (31. 05 2024). *HTML*. Abgerufen am 05 2024 von https://html.spec.whatwg.org/#the-mark-element

Wikibooks. (02. 11 2014). *Wikibooks*. Abgerufen am 05 2024 von https://de.wikibooks.org/wiki/Websiteentwicklung:_XHTML:_Geschichte