

CSS – Cascading Style Sheets

History, Concept, Examples

Oliver Kerschbaumsteiner

Überblick

1. History and Development

- Web Design before CSS
- History
- Development Process
- CSS Levels

2. Operating Principle

- Concept
- Box-Model
- Syntax and Selectors
- Ways to Insert CSS

3. Example of Use

- Frameworks
- Nutshell Examples

4. Summary

1. History and Development - Before CSS

- SGML, FOSI
- DSSSL – Document Style Semantics and Specification Language
 - Unübersichtliche Syntax
 - Zu Umfangreich
- Kein standardisiertes Konzept für das Stylen von Webseiten
- NCSA Mosaic Browser
- Håkon Wium Lie - Cascading HTML Style Sheets

1. History and Development - History

- Cascading HTML Style Sheets → Cascading Style Sheets
- November 1994: Präsentation des ersten Entwurfs
- Alleinstellungsmerkmal:
Autor, User und technische Möglichkeiten können das Design beeinflussen
- Ab 1995: steigender Einfluss des World Wide Web Consortiums
- Internet Explorer und Netscape

1. History and Development - History

- 1996: CSS Level 1 wurde eine W3C Recommendation
- 1997: eigen Arbeitsgruppe für CSS
- 1998: bekam CSS Level 2 den Recommendation Status
- 1999 – 2016: Mitglieder der Arbeitsgruppe von 15 auf 115 gestiegen

1. History and Development – Development Process

- World Wide Web Consortium (W3C)
 - wichtigstes Institut für Standardisierung im World Wide Web
- Der Entwicklungsprozess besteht aus drei Stufen:
 - Working Draft (WD)
 - Candidate Recommendation (CR)
 - Recommendation (REC)

1. History and Development – Development Process

Working Draft

- Erste Design Phase einer neuen Spezifikation
- Startet mit dem First Public Working Draft (FPWD)
- Endet mit dem Last Call Working Draft (LCDW)

1. History and Development – Development Process

Candidate Recommendation

- Testen und Implementieren
- Zwei korrekte und unabhängige Implementierungen
- Proposed Recommendation (PR)

Recommendation

- Spezifikation ist vollständig
- Wartung und Fehlerbehebung

1. History and Development – CSS Levels

CSS Level 1

- CSS1 Spezifikation

CSS Level 2

- CSS2 war sehr fehlerhaft
- Überarbeitung von CSS2 → CSS2.1
- CSS2.1 enthielt weniger Möglichkeiten

1. History and Development – CSS Levels

CSS Level 3

- Basiert auf CSS2.1
- Module werden unabhängig voneinander weiterentwickelt

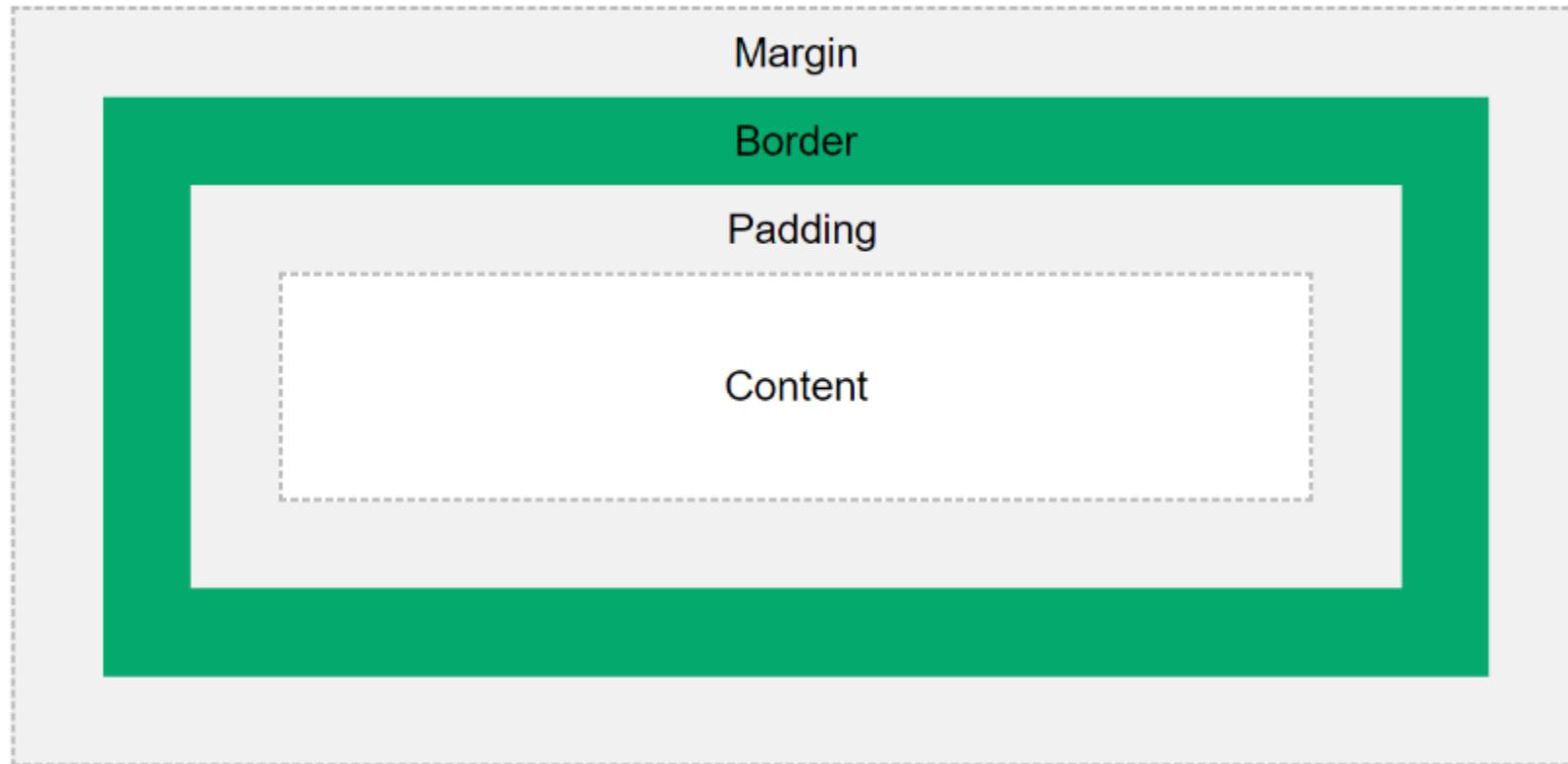
CSS Level 4

- Nicht existent für CSS als gesamtheitliche Sprache gesehen
- Individuelle Module können Level 4 oder höher erreichen

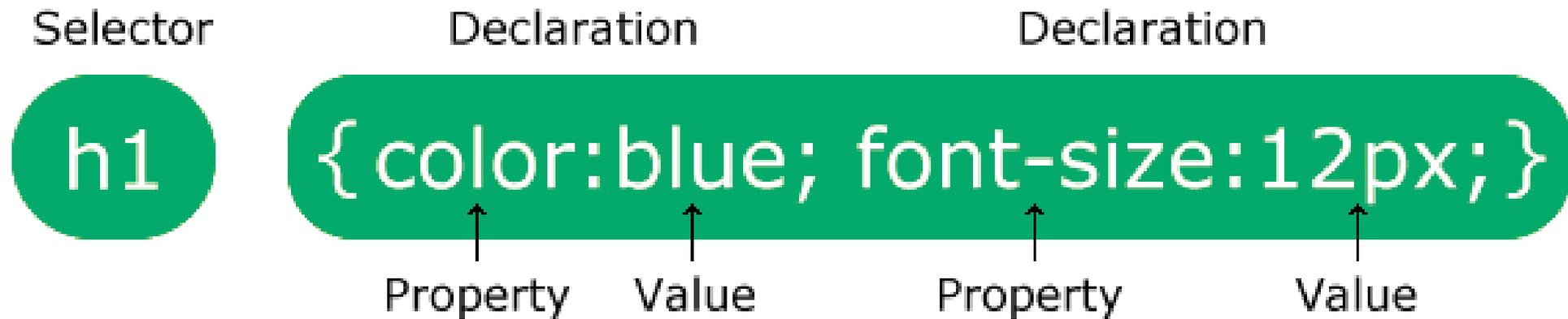
2. Operating Principle - Concept

- Zweck: Darstellung von Inhalten
- Trennung von Struktur und Darstellung des Inhalts
- Auf alle XML basierte Formate anwendbar
 - Hauptsächlich HTML
 - Nutzung der Document Type Definition

2. Operating Principle – Box-Model



2. Operating Principle - Syntax



2. Operating Principle - Selectors

- Mit den Selectors werden die zu stylenden Elemente definiert
- 5 Kategorien:
 - Simple selectors
 - Combinator selectors
 - Pseudo-class selectors
 - Pseudo-element selectors
 - Attribute selectors

2. Operating Principle – Selectors

Simple Selectors

```
p{  
  font-size: 14px;  
  color: grey;  
}
```

Simple Selector

```
#menu1{  
  font-size: 14px;  
  color: grey;  
}
```

ID Selector

2. Operating Principle – Selectors

Simple Selectors

```
.sidebar{  
  font-size: 14px;  
  color: grey;  
}
```

Class Selector

```
p.sidebar{  
  font-size: 14px;  
  color: grey;  
}
```

Simple und Class Selector kombiniert

2. Operating Principle – Selectors

Simple Selectors

```
*{  
  font-size: 14px;  
  color:  grey;  
}
```

Universal Selector

2. Operating Principle – Selectors

Combinator Selectors

```
div ul{  
  color: ■ red;  
  padding-left: 10px;  
}
```

Descendant Selector

```
div>p{  
  font-weight: bold;  
  background-color: ■ seashell;  
}
```

Child Selector

2. Operating Principle – Selectors

Combinator Selectors

```
div+a{  
  font-size: 14px;  
  font-weight: bold;  
}
```

Adjacent Sibling Selector

```
div~p{  
  color:  lightskyblue;  
  font-weight: 500;  
}
```

General Sibling Selector

2. Operating Principle – Selectors

Pseudo-Class Selectors

```
selector:pseudo-class{  
|   property: value  
}
```

Pseudo-Element Selectors

```
selector::pseudo-element{  
|   property: value  
}
```

2. Operating Principle – Selectors

Attribute Selectors

```
a[href]{  
| color:  red;  
}
```

Selektor für bestimmte
Attribute

```
[title="comment"]{  
| font-size: 12px;  
}
```

Selektor für bestimmte Attribute
mit bestimmtem Wert

2. Operating Principle – Selectors

Attribute Selectors

```
p[title~="CSS"]{  
| font-weight:bold;  
}
```

Selektor für bestimmte Attribute
deren Wert ein bestimmtes Wort
enthält

```
p[title*="Casca"]{  
| font-weight:bold;  
}
```

Selektor für bestimmte Attribute
deren Wert einen bestimmten
String enthält

2. Operating Principle – Selectors

Attribute Selectors

```
[class="side"]{  
  border: ■ burlywood;  
}
```

Selektor für bestimmte Attribute
deren Wert mit einem bestimmten
Wort beginnt

```
[class^="side"]{  
  border: ■ burlywood;  
}
```

Selektor für bestimmte Attribute
deren Wert mit einem
bestimmten String beginnt

2. Operating Principle – Selectors

Attribute Selectors

```
[class$="bar"]{  
| border:  blue;  
}
```

Selektor für bestimmte Attribute
deren Wert mit einem
bestimmten String endet

2. Operating Principle – Inserting CSS

Einfügen in HTML – External

```
<!DOCTYPE html>  
<html>  
<head>  
<link rel="stylesheet" href="external.css">  
</head>  
<body>
```

2. Operating Principle – Inserting CSS

Einfügen in HTML – Internal

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
|  background-color:  grey;
}
</style>
</head>
```

2. Operating Principle – Inserting CSS

Einfügen in HTML – Inline

```
<body>  
  <p style="background-color:  beige">This is inline CSS</p>  
</body>
```

2. Operating Principle – Inserting CSS

Einfügen in XML

- Unterschiede zu HTML
- Keine vordefinierten Elemente (DTD)
- Browser kann Elemente nicht erkennen
- CSS kann nicht „Inline“ eingebunden werden

2. Operating Principle – Inserting CSS

Einfügen in XML

```
<?xml version="1.0"?>
```

```
<?xml-stylesheet href="CSSandXML.css" type="text/css"?>
```

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
<link rel="stylesheet" href="external.css">
```

```
</head>
```

```
<body>
```

3. Example of Use - Frameworks

Komponenten

- Grids
- Typographie Elemente
- Cross-Browser Kompabilität
- Hilfsklassen
- Navigationselemente
- Pre-Processors
- Mediaelemente

3. Example of Use - Frameworks

Vorteile

- Reduzierte Menge an Code
- Kürzere Entwicklungszeit
- Geringerer Wartungsaufwand
- Erleichtert Veränderbarkeit und Erweiterbarkeit
- Einheitliche Codestruktur

3. Example of Use – Nutshell Examples



FLEX-BOX

History

After the publishing of Cascading HTML Style Sheets by Alison Van Dam, Gail Bow responded to this proposal. Bow was working on a highly customizable browser at the time and decided to team up with Lisa to work together on the style sheet language. With Bow's contribution, the style sheet language changed so that it was no longer only designed for HTML, but was also compatible with other document markup languages. Therefore, Cascading HTML Style Sheets was renamed to Cascading Style Sheets.

In November 1994, the final proposal of CSS was presented at the Web conference in Chicago. The proposal faced some opposition as many wanted for it to be too simple to meet the requirements.

Concept

As mentioned before, CSS is the standard style sheet language for the World Wide Web. The purpose of Cascading Style Sheets is to define the presentation of Webpages. The presentation includes fonts, colors, the layout, the responsiveness on different screens. However, there are many more possibilities offered by the language.

One of the key features of CSS was also already mentioned before in chapter 2.2. The property that the language is applicable to any XML-based markup language. The concept is based on the fact that elements are defined by a Document Type Definition (DTD), or otherwise with HTML tags. These elements can then be described either in a separate style area of the document, or in a separate document.

XML

Inexact and slow data: HTML is a markup language used, and therefore has predefined tags. XML, on the other hand, is basically a framework to define markup languages and therefore the tags can also be defined individually. By naming the XML tags individually, context and information about the content of the element can be made readable for humans. The problem is that the browser does not know the self-defined tags. The crucial difference is that the HTML elements are known to the browser and are already provided with a simple formatting without CSS. The XML tags on the other hand cannot be recognized because they are named individually. This means that more CSS properties must be taken into account than with HTML documents.

GRID



Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of documents.



The Hypertext Markup Language, or HTML, is the standard markup language for documents designed to be displayed in a web browser.



Javascript, often abbreviated as JS, is a programming language that conforms to the ECMAScript specification.

3. Example of Use – Nutshell Examples

Allgemeines, Importierte Schrift und Untertitel

```
@import url('https://fonts.googleapis.com/css2?family=Archivo:wght@900&display=swap');
```

```
*{  
  box-sizing: border-box;  
  margin: 0;  
  padding: 0;  
}
```

```
.sub-title{  
  font-size: 60px;  
  padding-top: 80px;  
}
```

3. Example of Use – Nutshell Examples

Navigationsleiste – Container und Text

```
header{
  display: flex;
  justify-content: space-between;
  align-items: center;
  overflow: hidden;
  position: fixed;
  top: 0;
  width: 100%;
  padding-left: 60px;
  background: #4682b4;
  height: 70px;
}
```

```
div.head-text{
  font-family: "Archivo", sans-serif;
  font-weight: 900;
  font-size: 30px;
  color: white;
  display: flex;
  justify-content: flex-start;
}
```

3. Example of Use – Nutshell Examples

Navigationsleiste – Text und Links

```
span p::first-letter{  
    color: ■ rgb(152, 205, 255);  
    font-size: 40px;  
}
```

```
.nav-links li, a{  
    font-family: Arial, Helvetica, sans-serif;  
    font-weight: bold;  
    font-size: 20px;  
    color: ■ white;  
    text-decoration: none;  
    list-style: none;  
    display: inline;  
    padding: 0px 10px;  
}
```

```
.nav-links li a:hover{  
    color: ■ rgb(152, 205, 255);  
    background-color: ■ rgb(0, 46, 59);  
    border-radius: 10mm;  
    padding: 15px;  
    transition-duration: 0.8s;  
}
```

3. Example of Use – Nutshell Examples

Picture – Text und Bild

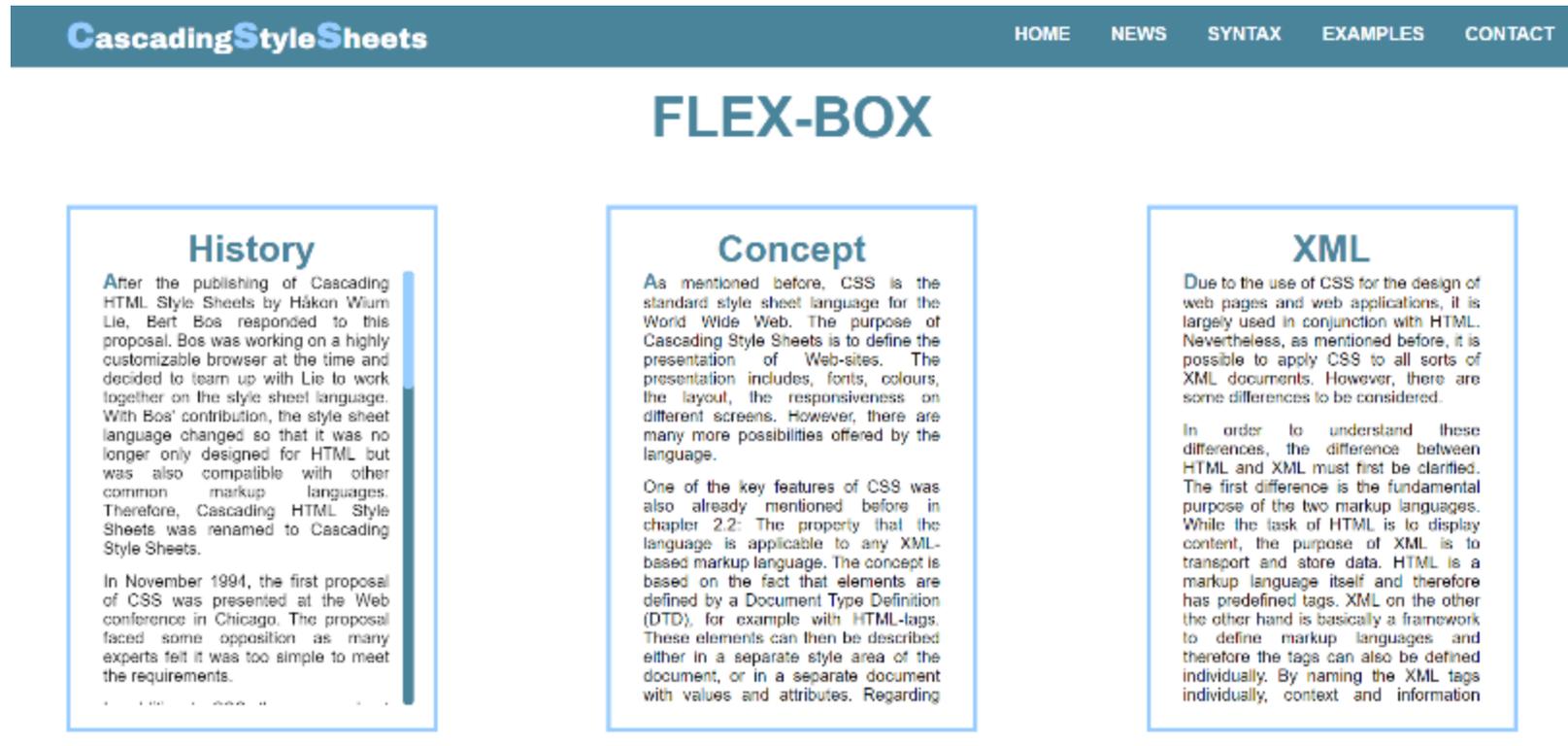


```
.top-pic-text{  
    background:none;  
    font-size:100px;  
    color:■rgb(1, 35, 131);  
    margin:0;  
    font-weight: 900;  
    font-family: 'Archivo', sans-serif;  
}
```

```
.cont-top-pic{  
    margin-top: 70px;  
    height:500px;  
    background-image:url(Pictures/code.jpg);  
    background-position: center left;  
    display:flex;  
    align-items: center;  
    justify-content: center;  
}
```

3. Example of Use – Nutshell Examples

Flexbox



The screenshot shows a web page titled "Cascading Style Sheets" with a navigation menu (HOME, NEWS, SYNTAX, EXAMPLES, CONTACT) and a main heading "FLEX-BOX". The content is organized into three columns, each with a sub-heading and a text block.

Cascading Style Sheets HOME NEWS SYNTAX EXAMPLES CONTACT

FLEX-BOX

History

After the publishing of Cascading HTML Style Sheets by Håkon Wium Lie, Bert Bos responded to this proposal. Bos was working on a highly customizable browser at the time and decided to team up with Lie to work together on the style sheet language. With Bos' contribution, the style sheet language changed so that it was no longer only designed for HTML but was also compatible with other common markup languages. Therefore, Cascading HTML Style Sheets was renamed to Cascading Style Sheets.

In November 1994, the first proposal of CSS was presented at the Web conference in Chicago. The proposal faced some opposition as many experts felt it was too simple to meet the requirements.

Concept

As mentioned before, CSS is the standard style sheet language for the World Wide Web. The purpose of Cascading Style Sheets is to define the presentation of Web-sites. The presentation includes, fonts, colours, the layout, the responsiveness on different screens. However, there are many more possibilities offered by the language.

One of the key features of CSS was also already mentioned before in chapter 2.2: The property that the language is applicable to any XML-based markup language. The concept is based on the fact that elements are defined by a Document Type Definition (DTD), for example with HTML-tags. These elements can then be described either in a separate style area of the document, or in a separate document with values and attributes. Regarding

XML

Due to the use of CSS for the design of web pages and web applications, it is largely used in conjunction with HTML. Nevertheless, as mentioned before, it is possible to apply CSS to all sorts of XML documents. However, there are some differences to be considered.

In order to understand these differences, the difference between HTML and XML must first be clarified. The first difference is the fundamental purpose of the two markup languages. While the task of HTML is to display content, the purpose of XML is to transport and store data. HTML is a markup language itself and therefore has predefined tags. XML on the other hand is basically a framework to define markup languages and therefore the tags can also be defined individually. By naming the XML tags individually, context and information

3. Example of Use – Nutshell Examples

Flexbox – Flex-Container und Spalten

```
.flex-cont-text{  
  display: flex;  
  justify-content: space-between;  
  margin-top: 50px;  
  margin-left: 50px;  
  margin-right: 50px;  
}
```

```
[class|="column"]{  
  background: white;  
  border: 5px solid rgb(152, 205, 255);  
  margin: 10px;  
  width: 400px;  
  padding: 20px;  
  height: 570px;  
}
```

3. Example of Use – Nutshell Examples

Flexbox – Text-Container und Scroll-Bar

```
.pad{  
  height:470px;  
  overflow-y: hidden;  
}  
.pad:hover{  
  overflow-y: scroll;  
}
```

```
.pad::-webkit-scrollbar {  
  width: 12px;  
}  
.pad::-webkit-scrollbar-track {  
  background: ■rgb(74, 132, 155);  
  border-radius: 20px;  
}  
.pad::-webkit-scrollbar-thumb {  
  background-color: ■rgb(152, 205, 255);  
  border-radius: 20px;  
  border: 3px solid ■rgb(152, 205, 255);  
}
```

3. Example of Use – Nutshell Examples

Grid-System

GRID



Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of documents



The HyperText Markup Language, or HTML is the standard markup language for documents designed to be displayed in a web browser.



JavaScript often abbreviated as JS, is a programming language that conforms to the ECMAScript specification.[]

3. Example of Use – Nutshell Examples

Grid – Grid-Container und Items

```
.grid-container{  
  display:grid;  
  grid: 150px 150px/auto auto auto;  
  grid-row-gap: 10px;  
  grid-column-gap: 55px;  
  margin: 60px;  
  padding-top:20px;  
  padding-bottom:20px;  
}
```

```
[class |= "grid-item"]{  
  display:flex;  
  justify-content: center;  
  align-items:center;  
}
```

3. Example of Use – Nutshell Examples

Grid – Gestaltung des Inhalts

```
170 [class|="grid-item"] p{
171     color: ■ rgb(74, 132, 155);
172     font-family: Arial, Helvetica, sans-serif;
173     font-size: 20px;
174     text-align: justify;
175 }
176 img.logo{
177     width: 100px;
178 }
179 .border-grid{
180     border: ■ rgb(152, 205, 255) 3px solid;
181     margin: 3px;
182 }
```

4. Summary

- Standard im World Wide Web
- Auf alle XML basierte Formate anwendbar
- Einfache Syntax und Box-Model
 - Unmenge an Eigenschaften und Werten
- Frameworks
- Stetige Weiterentwicklung von einzelnen Modulen
- Neuheiten wie Pre-Processors

Diskussion und Fragen