1. *UI and UX design*

The Acronyms UI and UX are quite common in modern web discussions – is is like the basic scaffolding of a modern webpage or webapp. These Acronyms representing the words *User Interface* and *User Experience Design*.

1.1. *User Experience Design*

But what is UX? If you hear the word User Experience Design you probably think about arranging objects or designing colours and graphics on the screen. But User Experience is quite different. It is more about the experience of using things. The aim is to find the easiest way of communication which is intuitive and simple. But getting on the right lane is a long progress in getting in touch with people. Knowing the target customers is the responsibility in UX design. It is important to know their habits and reactions to face their problems. Design is not about arranging objects in an aesthetically way. It is a progress of solving problems. We are all facing problems everywhere and for the most cases there might be a solution out there. But life is not a pre written script. It is unpredictable. There are so many edge cases out there and the goal of a good UX design is to solve as many problems as possible.[1]

> see http://blog.careerfoundry.com/ux-design/the-ux-design-process-an-actionable-guide-to-your-first-job-in-ux

> see Levin, Michal: Designing Multi-Device Experiences – An Ecosystem Approach To User Experiences Across Devices. (First release): O‘Reilly and Associates, 2014, S. 20 –

1.2. *User Interface Design*

When we are talking about User Interface Design, it is not a conversation about subjective art. On the contrary it is an objective principled form of human conversation and behaviour. It is like Steve Jobs said: “Design is not just what it looks like and feels like.

[1] see http://uxmag.com/articles/explaining-ux-design-to-your-team

Design is how it works" (Steve Jobs). But in spite of this User Interface is based on a good UX design. These two Principles are connected to each other. It is like an environment [2, 3, 4]

Design Principles

Design Principles are the language and the environment of expressing what is right and wrong in designs. It expresses a state so that the user has the best experience at handling the design. In Web Design there is everything about usability. The main focus of web design is to archive a visitor with a long lasting attention. The user should be interested in the topics and should get the information straight forward, without clicking through unpredictable dialogs or sections. These principles are the rule sets and the scientific laws of the usability, like the gravity law on earth. They were crafted during a long period of studying humans' behaviour and understanding the environment. That is the reason why you should design for the people and not for the aesthetic.

»People ignore design that ignores people.«
– Chimero, Frank

1.2.1. The Principle of Proximity

The principle of proximity is based on the perception of the distance between objects. Simplified it references to the grouping principle. Two objects, that are closer together pretend to have a relationship. In comparison, two objects that are further apart would have less relation. In general it is a pattern system, that describes a function. Ordering objects according to each other based on the function leads to a comprehensible layout and design. With this little enhancement it is possible to generate a huge impact in improving the design. Think about frustrated and confused users, who are not able to

[2] Krishna, Golden (2016). The Best Interface is No Interface. In: SlideShare. [online] City: Golden Krishnas presentation at From Business to Buttons in Stockholm. Available at: https://de.slideshare.net/inusese/golden-krishna-the-best-interface-is-no-interface

[3] Vimeo Krishna, Golden – The Best Interface is No Interface: https://vimeo.com/48629451

[4] McKay, Everett. (2014). Ui is Communication: How to design intuitive, user-centered interfaces by focusing on effective communication. In: SlideShare. [online]. Available at: ttps://de.slideshare.net/EverettMcKay/ui-is-communication-how-to-design-intuitive-usercentered-interfaces-by-focusing-on-effective-communication [10.03.2017].

get any kind of information in case of misleading arrangement of objects and how they are organized. The solution is a clear and well-considered layout that puts less strain on users to find the information they need.[5]

![](../../images/figure-1-1_example_of_the_proximity_principle.png)

Example of the proximity principle

1.2.2. Visual Feedback and Visual Prominence

1.2.3. unreduzierbare Einfachheit

> “Deciding what not to do is as important as deciding what to do.”

> – Steve Jobs

1.2.4. Intuition

1.2.5. Funktionalität vor Ästhetik

1.2.6. Innovation

Der Erfolg von Innovation ist wie eine Schaukel zwischen Langeweile (nichts neues) und Argwohn (zu viel neues)

1.2.7. Form and Colorcomposition

1.2.8. Repeatable patterns

> see https://www.smashingmagazine.com/2015/12/design-principles-to-evaluate-your-product/

1.3. The relation of impact and effort

> see https://www.smashingmagazine.com/2016/10/giving-your-product-a-soul/)

1.4. Googles Material Design principle

1.4.1. The minimalistic approach

1.5. The thumb zone – Mobile Design

> see https://www.smashingmagazine.com/2016/09/the-thumb-zone-designing-for-mobile-users/

[5] see Lowdermilk, Travis: User-Centered Design. (First release) – 1005 Gravenstein Highway North, Sebastopol, CA 95472: O’Reilly Media, Inc., 2013, S. 63 – 64

2. *Das Layout*

2.1. *Erklärungen*

2.1.1. *Media Queries*

Media Querys sind Abfragen, die es dem Medium *(in diesem Fall dem Browser)* ermöglichen zu ermitteln welche Breite der sogenannte ***view-port*** *(Innere Bereich eines Browserfensters)* hat. Anhand dieser Media queries werden dann Syleregeln angewendet. Eine Media Query verfügt auch über die Optionalität die Ausrichtung eines Geräts, wie Hochformat oder Querformat, oder auch die Pixeldichte *(für Retinageräte)* abzufragen.

2.1.2. *Breakpoints*

Breakpoints sind bestimmte Punkte oder Punktespannen, gemessen in px, welche bestimmte Bereiche definieren. Diese Bereiche werden dann auf die Media Queries angewandt, um eine bestimmte Layoutregel darauf anzuwenden.

Die folgenden Breakpoints wurden bei der Website verwendet um die folgenden 5 Gerätegrößen abzudecken.

von px	bis px	Gerät
0	510	Mobile Devices
511	768	Tablets
769	1200	Laptops
1201	1440	Desktops
1441	∞	Ultra Wide

2.2. *wie alles anfing*

(Fixed Layouts und der Beginn des Webs)

2.3. *Responsive Layouts*

2.4. *Fluid oder liquid layouts*

Der Begriff fluid oder liquid Layout kommt davon, dass dieses ohne fixen Breitenangaben auskommt. Das heißt alles wird in Prozent gerechnet und behält dadurch seine Proportionen bei Skalierung des Browserfensters. Das klassische Beispiel für ein Liquid Layout wäre die Einteilung von Sidebar und Main Bereich, wo der Hauptteil eine Breite von 70% hat und die Sidebar eine Breite von 30%.

Diese Technik kommt jedoch oft mit der Kombination von Media Queries vor, da diese ermöglicht andere Prozentuale Verhältnisse für mobile Geräte zu definieren. Im Beispiel der Sidebar würde das bedeuten, dass diese auf mobilen Geräten eine Breite von 100% erreichen würde und sich unter den Hauptteil fügen würde. Dieses flüssige behalten ist der Hauptgedanke eines Fluid Layouts. Es beschreibt den Fluss der Elemente auf verschiedenen Endgeräten. Dieses Phänomen kann man nicht nur beim Vergrößern des Browserfensters begutachten, sondern auch beim Ändern der Geräteausrichtung, von Horizontal auf Vertikal oder vice versa.

> vgl. Implementing Responsive Design p.25

2.5. *Elastic Layouts*

2.6. *Hybrid Layouts*

Die letzte und Layoutoption ist das Hybride Layout. Es kombiniert die Eigenschaften der vorangehenden Beispiele.

Dies Kommt zum Beispiel bei unserem Grid Layout vor. Da der Abstand der Grid Spalten, die *grid-gap* in px angegeben ist. Dadurch dass ein sogenanntes 12er grid (bestehend aus 12 Spalten) im Einsatz ist passt sich die Breite einer einzelnen Spalte dem restlichen Platz an. Zusätlich steuern noch Media Queries mit einem Breakpoint für mobile Endgeräte die *grid-gap*, damit diese auf Smartphones zum Beispiel geringer ist.

The last Layoutoption is the hybrid layout. It combines the abilitys of the previous examples.

3. Das Gridsystem

Eines der wichtigsten Gestaltungsprinzipien im Web-Design ist ein Grid System auf dem alles Basiert. Dieses Layout

4. Mobile First – but why?

> ”Es gibt mehr Ding im Himmel und auf Erden, als Eure Schulweisheit sich träumt“

> – Shakespeare Hamlet

Diese Worte spricht Hamlet zu seinem Freund Horatio, doch diese Worte haben nicht an Aktualität verloren – im Gegenteil. Denn der durchschnittliche User, der eine Website aufruft hat kein MacBook Pro der letzten Generation und das neueste iPhone und das schnellste iPad. Die Verantwortung nicht für sich selbst zu designen sondern den durchschnittlichen end-user im Kopf zu behalten ist ein durchgängige Thematik.

Die „World Bank“ berichtet, dass mehr als 1.1 Milliarden Menschen auf der Welt zugang zu High Speed internet haben. 3.2 Milliarden Menschen haben Internetzugang mit mäßiger Geschwindigkeit und 5.2 Milliarden Menschen besitzen überhaupt ein Mobile Telefon. 7 Milliarden Menschen leben mit einem mobilen Internet Zugang.

> Enzenhofer, Klaus. (2017) UX reports from the trenches: Topconf Linz. 2017:

> [Smashing Mag Artikel](https://www.smashingmagazine.com/2017/03/world-wide-web-

not-wealthy-western-web-part-1/)

5. The component based system

6. The technological Stack

Technology is one of the most important parts when it comes to the point of development and production. It is a combat between using a bulletproof solution, that might be outdated, and using the fanciest and latest technologies, which are often poorly tested or not production ready. The problem is the speed, technologies are changing and updating. It is like a competition against the time. Every week there is a new major version releasing and things are going even faster now. Maybe when writing these lines a new framework is now in production somewhere out in the world. But what is the single and only solution? Short hint, there is no only solution. Think about all problems out there, but every problem might need a different solution. So there are no laws and rules how to tackle a specific problem it is more looking for the needs of the customer and combining them with the habits and preferences of the developers. Finally, the developer has to work and maintain these technologies. So in the ecosystem of the Altwiener Christkindlmarkt, the focus was on a modular CMS like Contentful and Storyblok. Instead of a static site generator or Wordpress. On the one hand, Wordpress would be pretty simple to install and a bunch of plugins would be available out there, but on the other hand, a CMS like Storyblok which is API driven is platform independent. The benefit of an API driven CMS is that you can choose your stack on your own. In the case of Storyblok, a JSON *(JavaScript Object Notation)* based *Application Programming* Interface is easier to handle in JavaScript as in PHP.

6.1. What is HTML5

If you talk about HTML5, you talk about a Hyper Text Mark Language short HTML. In general, it is like a structuring language for content. In other words, you describe the content. You tell the interpreter so that this line of code is a headline of the first section or this area is a sidebar. The interpreter, in many cases a modern web browser can handle these markup and render it. HTML5 is a superseding of HTML4 and all previous versions of HTML. It is kind of the *next generation of HTML*, so it is the scaffolding for a new way of web application and websites. To conclude HTML5 takes advantage of new features which are shifting the way of browser experience. To sum some up HTML5 is cross-platform compatible, so all you need is a modern web browser that can interpret these new tags and definitions. Despite the fact it is a new way of a formal documentation for unwritten standards in most browsers. So it was a shift for developers to rely on new features in a way that every browser interpret these features according to a standard. HTML5 brings some new features as well. Semantic elements

like `<header>`, `<footer>` or `<article>`. Geolocation, a persistent local storage, offline web applications or even microdata to extend the vocabulary of HTML5 with custom semantics.[6]

Turning to the point that you want to use HTML5 features on the web application. The question, can I use HTML5 on my web page is misleading, probably HTML5 is not one big thing, it is a collection of features and some browsers only supporting a variety of features, like certain tags or video elements. Obviously, it is possible to detect the support of those features. But getting familiar with the detection techniques it is necessary to get familiar with the Document Object Model short DOM.[7]

6.1.1. *What is the DOM*

So what is the Document Object Model? It is not the source code you write in a .html file. Therefore the DOM is the result of your source code parsed by the browser, even though it is not the source code if you click in your browser on View Source. As can be expected View Source shows you exact the HTML that makes up the page. If you want to have a look at the DOM you need to inspect the site with the developer tools. There you get a brief overview of all document nodes. You can click on them and expand them. So it is like a box model where every box is wrapped in another box. So if you want to manipulate the DOM during runtime of a page – in the case of user interaction – you have to take advantage of JavaScript. You can fetch some DOM-Nodes or Single Nodes with a simple `document.querySelector(‘container’);` This would return the first Node of type container. Now you can store this node in a variable and manipulate it.[8]

In other words, JavaScript is a language the browser can understand and interpret. On the contrary, there is the DOM – the place where the magic happens. So facing this two arguments you can say JavaScript is a DOM accessing and manipulation language where you can watch events, attached to some DOM nodes. In other words, you can give a button a click event that executes a function when the button is pressed.[9]

> “The (...) DOM is an application programming interface (API) (...) It defines the logical structure of documents and the way a document is accessed and manipulated.”
>
> *(W3C)*[10]

In point of detection techniques for HTML5 features, we are accessing the `window` or `navigator` Object and testing if the property for the particular feature is available in this context. For this kind of testing, there is an HTML5 detection library called Modern-

[6] cf. Pilgrim, Mark: HTML5 – Up and Running. (First Edition) – 1005 Gravenstein Highway North, Sebastopol, CA 95472: O’Reilly Media, Inc., 2010, p. 11 – 12.
[7] cf. Pilgrim, HTML5 – Up and Running, p.29
[8] COYIER, CHRIS: What is the DOM? – https://css-tricks.com/dom/, 10.03.2017
[9] COYIER, CHRIS: What is the DOM? – https://css-tricks.com/dom/, 10.03.2017
[10] W3C: What is the Document Object Model – https://www.w3.org/TR/DOM-Level-2-Core/introduction.html, 10.03.2017

```
<div class=“header”>
    <h1>My Header</h1>
</div>
```

```
<header>
    <h1>My Header</h1>
</header>
```

MDN: HTML element reference – https://developer.mozilla.org/en-US/docs/Web/HTML/Element izr. It is open source and under the MIT-licence and detects the support for HTML5 and CSS3 features. It is important that you include this script in the `<head>` Tag of the page. This is important because you need it before the DOM is built up so that you can check the support of the features. This is possible because page rendering is blocked until the script is loaded and executed.[11]

6.1.2. Using HTML5 tags and getting rid of the boxing hell

If you look back in the early beginnings of HTML and the web, HTML was only a structuring language like XML or maybe Markdown. It was like you had a site where you place your text in a word document and that was it. Of course, it was possible to change the background colour or the font family. But in the beginning of web HTML was not considered to be used for special layouts and arranging objects on the page. Although one essential feature like tables where used misleading and not the way they meant. Tables were considered for tables (nothing else), besides the fact that using tables is only a good option if you want to display a score table or something that fits into a table. However, you have to consider that tables in a responsive fluid layout are always a curse. But developers struggled a lot with the layout problem till the float was invented in CSS 2.1. Now it was possible to float containers and position them. The ultimate change came in the HTML version 4 where a new container occurred. The div element was born. HTML4 came with a lot of breaking changes and one of them was a normal box without any additional styling. It was like a LEGO Box, where it was possible to nest things deeper and deeper. As can be expected this was the point where maintainability of websites was even more crucial as before. The problem was the end of the closing divs. That was the point where you had a tree with 10 closing divs at the and – that was a pleasure to figure out which one was the correct one. In the same way, it was even harder for search engines or crawlers to get semantic information about the page. Obviously, all important information about a page was then put in the meta information in the head section of a page. In other words, people wrote a huge list of any information only in the keywords section to be better rated at google. In this case, it was hard to find the right page over a search engine because so much misleading information was given to the search robots. God blesses the invention of HTML5, now it was possible to build a semantic correct page and the Google-Bots could focus on retrieving the information directly from the page without concentrating on wrong keywords or meta descriptions. Furthermore, it was a bless for developers, as it was way easier to figure out what the content describes a header was a header tag and not a div with a class name. The following example shows the difference at the closing tag. The starting tag is maybe easier to read but the real benefit is the closing tag where you see immediately that the header is closed.

For structuring the page sections instead of div elements following new containers were created to enhance the semantic of the document. The `<section>` node was create to describe a section in a document. In a section, it is recommended to structure the text in

[11] cf. Pilgrim, HTML5 – Up and Running, p.30

IE	***Firefox***	***Safari***	***Chrome***	***Opera***	***iPhone***	***Android***
8.0+	3.5+	4.0+	4.0+	10.5+	2.0+	2.0 +

Can I use: Web Storage - name/value pairs – http://caniuse.com/#search=localStorage, 13.03.2017

```
// included in the <head> section of the page

var fontversion = ‘1’;
var fontcache = ‘fontcache-altwiener-markt’;

function addStyles(_styles) {
 var style = document.createElement(‘style’);
 style.innerHTML = _styles;
 var script = document.getElementsByTagName(‘script’)[0];
 script.parentNode.insertBe   fore(style, script);
}

try {
 var s = window.localStorage.getItem(fontcache);
 if (s !== null) {
  var parts = s.split(‘####’);
  if (parts.length !== 2 || parts[0] !== fontversion) {
   window.localStorage.removeItem(fontcache);
  } else {
   addStyles(parts[1]);
  }
 }
} catch (e) {
 console.warn(‘A Problem occurred by setting the Fontcache: ‘, e);
};
```

```
// included before the closing </body> tag of the page

 (function(ls, x) {
  try {
   if (ls.getItem(fontcache) === null) {
    var h = new x();
    h.open(‘GET’, ‘/styles/fonts.css’);
    h.send();
    h.onload = function() {
     addStyles(this.responseText);
     ls.setItem(fontcache, fontversion + ‘####’ + this.responseText);
    }
   }
  } catch (e) {
   console.warn(‘A Problem occurred by retrieving the Fontcache: ‘, e);
  }
 })(window.localStorage, XMLHttpRequest);
```

so-called `<article>` nodes. In contrast, a article node could contain a variety of section elements. These article nodes are describing a stand-alone content, that can be independently from the rest of the page. In addition, there was a tag for a sidebar `<aside>`, that is very common in blog or newspaper layouts. As well for navigations, the `<nav>` tag was invented that can only contain navigation links.[12]

6.1.3. *The HTML5 Storage*

Heading up to part of storing data it is necessary to know, that storing key/value pairs in browsers was a crucial thing before the introduction of the so-called Web Storage – some browsers are referring to it as the “Local Storage” or “DOM Storage”. Looking to the historical part of storing elements in the browser cookies seemed as a good solution in this days. On the one hand it seemed like a good idea storing data persistent even after navigating away from the site or closing the browser, but on the other hand, it is unlike with cookies. You have to send it manually if you want to transmit it to the remote web server. Considering the use of the local storage you need not be aware of using it, accordingly, the browser support is amazing if you check the Table.[13]

6.2. *CSS and SCSS*

When we are heading to the point of styling our page, CSS is the only solution. CSS *(Cascading Style Sheets)* is a language written on top of HTML to markup tags and control the layout and design of web page. If you were surfing the web in the year of 1995 and having a look at the websites of this period, you will probably recognize, that CSS was not invented yet. The pages consist out of a straight line orientated layout with any styles. Thank God now we have the third major version of CSS and we are facing the beta version of CSS4. Accordingly, we are hitting a point where CSS might be magical for you in fact of all possibilities. To begin let us start with the benefits CSS brought to a sluggish web page. First of all, with CSS, we have the choice to format text, paragraphs and headlines. You are able to change the font family, the size, the font weight and even the letter spacing – like we are used from print publishing tools like InDesign. In addition, it is possible to indent paragraphs or apply drop caps to them, in other words, you can style them like a magazine layout. The next useful point is with CSS you get the ability to arrange Objects – in our case divs and HTML5 tags – to provide a box model layout. In old days this was hacked with the float property – like the HTML tables, the float property was not meant for this, it was provided for text floats around images.[14]

Nevertheless, the web and particular CSS got the *Flexbox Layout* shortly flexbox. This new feature aims to be more efficient in the way to layout and align items with distributing space in a container. The most important benefit of this property is that it can be

[12] MDN: HTML element reference – https://developer.mozilla.org/en-US/docs/Web/HTML/Element

[13] cf. Pilgrim, HTML5 – Up and Running, p.128 – 129

[14] cf. Sawyer McFarland, David: CSS3 – the missing manual. (Third Edition) – 1005 Gravenstein Highway North, Sebastopol, CA 95472: O’Reilly Media, Inc., 2013, p. 1f

applied even if the size is unknown or dynamic. This is a huge benefit in responsive and fluid layouts and was a shift in so-called flexible Layouts. It is possible that containers expand automatically to fill free space or shrink them to prevent overflows. In other words, it provides a flexible layout for any kind of device or screen size. The main goal of flexbox was to arrange small-scale layouts within a grid component. Even though the CSS grid is in production and not supported in every browser – flexbox is used for grids as well in the moment. The flexbox properties are well supported now (Internet Explorer 11+ and all other major browsers) and giving us a new dimension in arranging objects in the way it is meant to.[15]

6.2.1. *Anatomy of a Style Declaration*

To understand the functionality of CSS it is important to now that CSS is not a programming language. Instead, it is a declaration based markup, that describes with a selector and a declaration block, which adjectives should be applied on these elements, selected with the selector. Often this kind of style declaration is easy to understand, as it is similar to English, for example, take a look at following style rule. `p { color: black, font-family: Henriette,  font-size: 1em }`. This example describes the properties for a standard paragraph, “Make the text black and take the font Henriette and make it 1em tall”. So It is pretty clear what is going on there.[16]

If you take a deeper look at the previous declaration you will see that it is built out of certain components. First of all, we need the selector, that tells the browser which element or elements on a page to select – in our case, it was a paragraph. The next element is the declaration block, you can identify it with an opening curly brace and with a closing curly brace. Inside this declaration block, you will see the list of declarations for the selected element. A declaration consists out of a key value pair. As can be expected, a Property like `font-size` and a value.[17]

6.3. *The benefits of a static typed Language*

6.4. *Automation with Gulp*

[15] cf. Coyier, Chris: A complete Guide to Flexbox – https://css-tricks.com/snippets/css/a-guide-to-flexbox/, 16.03.2017

[16] cf. Sawyer McFarland - CSS3 – the missing manual, p.36 – 37

[17] cf. Ebd., p.37

7. Search Engine Optimization (SEO)

7.1. Content Delivery Network auch CDN

7.2. Cannonical Links

7.3. XML Sitemap

7.4. Imageservice

7.5. Optimierung Above the Fold

7.6. Browser Caching

7.7. Korrekte Anwendung mittels E-Tags