Tomáš Mikula

Uživatelská rozhraní

Katedra informatiky, Přírodovědecká fakulta, Univerzita Palackého v Olomouci

Copyright © 2022 Katedra informatiky, Univerzita Palackého v Olomouci
www.inf.upol.cz

Tento text slouží jako doplňkový učební materiál pro kurz *Uživatelská rozhraní*, jenž je vyučován v rámci bakalářského studia na Katedře informatiky Přírodovědecké fakulty Univerzity Palackého v Olomouci. Kurz je zaměřen na problematiku návrhu uživatelských rozhraní s důrazem na rozhraní softwarových aplikací. Čtenář získá základní znalosti a dovednosti potřebné pro tvorbu kvalitních a dobře použitelných uživatelských rozhraní. Text je průběžně rozšiřován a doplňován. Poslední změna proběhla 10. března 2022. Pokud v textu naleznete chyby, prosím, kontaktujte mne prostřednictvím e-mailové adresy mail@tomasmikula.cz.

Děkuji Martinu Trnečkovi za jeho připomínky a cenné podněty ke kvalitě textu.

Tomáš Mikula

Obsah

Úvod **9**

Stručná historie **11**
První počítače . . . 11
Dávkové zpracování . . . 11
Příkazový řádek . . . 12
Textové uživatelské rozhraní . . . 13
Grafické uživatelská rozhraní . . . 14
WIMP paradigma . . . 15
Xerox Alto . . . 15
Atari 2600 . . . 16
Xerox Star . . . 16
Apple Lisa a Macintosh . . . 17
Microsoft Windows 1.0 až 3.1 . . . 17
Microsoft Windows 95 . . . 18
Nokia 2110 . . . 18
Palm (PDA) . . . 19
Post-WIMP paradigma . . . 19
Apple iPhone . . . 20
Amazon Kindle . . . 21
Apple iPad . . . 22
Kinect . . . 24
Natural-language user interface (NLUI) . . . 24
Chytré hodinky . . . 24
Oculus Rift . . . 24
Playstation 4 - DualShock 4 . . . 25

Design a psychologie **27**
Výzva designu ve vývoji softwaru . . . 27
Human-centered design (HCD) . . . 28
Afordance . . . 28
Signifier . . . 29
Mapování . . . 31
Zpětná vazba . . . 31
Omezení . . . 33
Mentální (konceptuální) model . . . 35
Myšlení rychlé a pomalé . . . 36

Konceptuální model lidského chování . . . 37
Sedm fází akce . . . 38
Úrovně lidského vnímání . . . 39
Lidská paměť . . . 41
Znalosti v hlavě . . . 41
Znalosti ve světě . . . 42
Millerovo magické číslo . . . 46
Zákony uživatelského rozhraní . . . 47
Efekt estetické použitelnosti . . . 47
Doherty threshold . . . 47
Fittsův zákon . . . 47
Hickův zákon . . . 48
Jakobův zákon . . . 48
Zákon výstižnosti . . . 48
Millerův zákon . . . 49
Occamova břitva . . . 49
Paretovo pravidlo . . . 49
Parkinsonův zákon . . . 50
Postelův zákon . . . 50
Efekt pozice v posloupnosti . . . 50
Teslerův zákon . . . 50
Von Restorff efekt . . . 51
Zeigarnik efekt . . . 51
Gestaltismus – zákon seskupení . . . 52
Gestaltismus – zákon blízkosti . . . 52
Gestaltismus – zákon podobnosti . . . 52
Gestaltismus – zákon o vizuální spojitosti . . . 53

Grafické uživatelské rozhraní **55**
Grafický layout . . . 55
Dva způsoby prohlížení: čtení a skenování . . . 56
Uživatelé jsou netrpěliví . . . 57
Návrh grafického rozložení . . . 57
Principy tvarové psychologie . . . 61
Mobile-first design . . . 64
Typografie . . . 68
Font vs. Typeface . . . 69
Mean vs. Base line . . . 69
Size, weight, height . . . 69
Ascender vs. Descender . . . 70
Whitespace . . . 70
Alignment . . . 70
Tracking vs. Kerning . . . 71
Leading . . . 72
Serif vs. Sans-serif . . . 72
Navigace . . . 74
Vrstvy aplikace . . . 74

Typy navigace 74

Prvky grafického uživatelského rozhraní **79**
Příkazy a akce 79
Button 79
Navigation bar 82
Tool bar 85
Status bar 87
Search bar 89
Tab bar 89
Progress bar/indicator 90
Drop-down list 90
Dialog 91
Uživatelský vstup 92
Label 92
Text input 93
Picker 95
Switch 96
Checkbox 97
Radio button 99
Složené prvky 99
List 99
Collections 100
Card 100

8 Uživatelská rozhraní

Úvod

„The next big thing is the one that makes the last big thing usable.“
Blake Ross, Co-creator of Mozilla Firefox

„If you think good design is expensive, you should look at the cost of bad design.“
Dr. Ralf Speth, Chief Executive Officer of Jaguar Land Rover

Uživatelská rozhraní jsou všude okolo nás. Těch kvalitně navržených si většinou ani nevšimneme. V případě těch méně kvalitních mnohdy ztratíme několik hodin času bez jasného výsledku.

Co rozhoduje o kvalitě uživatelského rozhraní? Jak vlastně lidé interagují s objekty? Na tyto otázky se pokusíme odpovědět v následujících kapitolách.

Průvodce studiem
Na začátku je nutné zdůraznit, že základy designu uživatelského rozhraní zvládne opravdu každý, není nutné disponovat talentem nebo citem. Základní přehled by měl mít každý, kdo se jakkoli podílí na tvorbě software, který má být používán lidmi.

10 Uživatelská rozhraní

Stručná historie

V této kapitole se budeme věnovat stručné historii uživatelských rozhraní. Za uživatelské rozhraní budeme považovat jakýkoli způsob interakce člověka se strojem nebo objektem. Z toho důvodu v kapitole zmiňujeme jak historii hardware tak software.

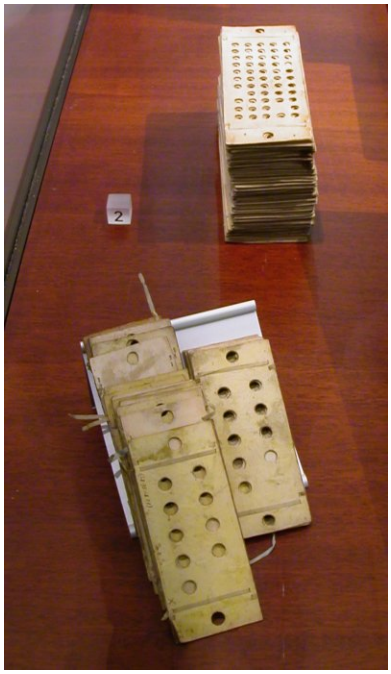
První počítače

Charles Babbage v roce 1837 přichází s návrhem obecně použitelného mechanického počítače – *Analytický stroj*. Uživatelský vstup byl zprostředkován pomoci *děrných štítků*. Pro výstup byla použita tiskárna, plotter a zvonek.
V roce 1945 ve svém článku *As We May Think* přichází Vannevar Bush s konceptem zařízení zvaném *Memex*. Zařízení mělo umožnit „zvětšení lidské paměti". Návrh popisoval stroj ve formě pracovního stolu[1] s několika obrazovkami a klávesnicí. Hlavním účelem mělo být ukládání/čtení informací ve formě asociací – podobně jako lidský mozek.
Koncept Memex významně ovlivnil vývoj hypertext systémů jako je například *World Wide Web (WWW)* a systémů známých pod názvem *Personal knowledge base* sloužících k uchování, reprezentaci a následné získání informací od uživatele.

Dávkové zpracování

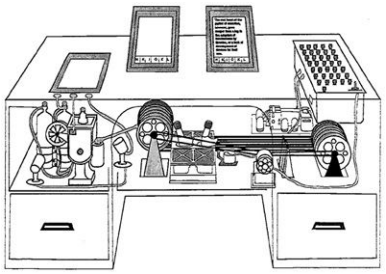
Okolo roku 1950 se začínají objevovat sálové počítače (také známé pod názvem *mainframe*). Provoz počítačů byl natolik nákladný, že pro jejich efektivní využití bylo potřeba zadávat vstup v takzvaných *dávkách*. Každá dávka se skládala ze sady *děrných štítků* (případně papírových pásek). K tomuto účelu sloužily speciální stroje – *děrovače* – podobné psacím strojům. Výstup zprostředkovávala řádková tiskárna. Mezi člověkem a počítačem neprobíhala žádná interakce v reálném čase a v případě chyby na vstupu nedocházelo k žádné okamžité zpětné vazbě.
Roku 1963 Ivan Sutherland prezentuje v rámci svého doktorského studia revoluční software s názvem *Sketchpad*. Jednalo se o první

Obrázek 1: Děrné štítky používané analytickým strojem.

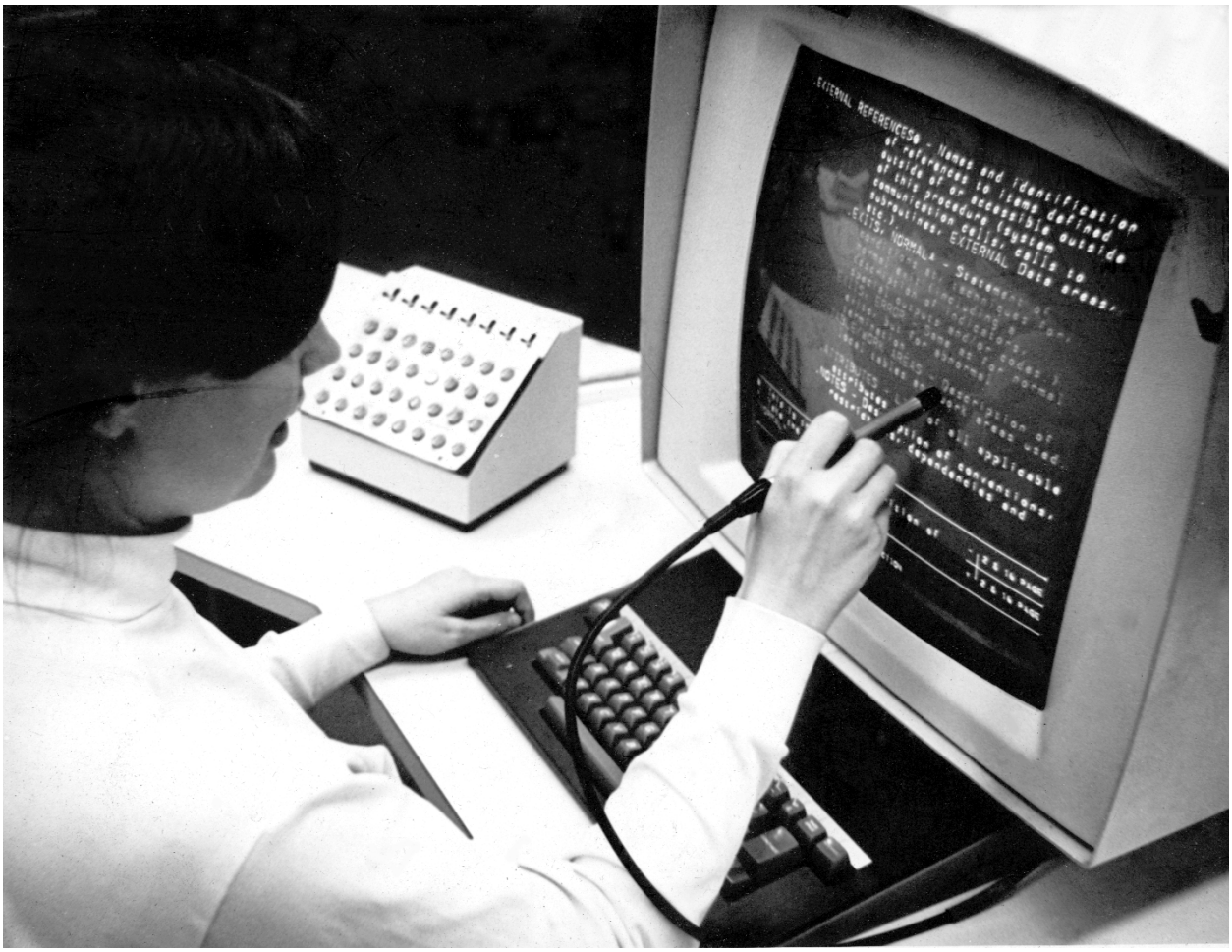


[1] Všimněme si, že ani v teoretickém konceptu si nikdo nedokázal představit zařízení velikosti hodinek.

Obrázek 2: Koncept zařízení Memex.



interaktivní systém v době dávkového zpracování. Systém využíval *light pen* – speciální světelné pero, které uživateli umožňovalo ukazovat na daný bod na CRT obrazovce.



Obrázek 3: Demonstrace zařízení Light-pen.

Příkazový řádek

Rozhraní příkazové řádky neboli *Command line interface (CLI)* bylo vyvinuto na základě monitorů dávkového zpracování, jeho odezva však byla řádově rychlejší (došlo ke zrychlení z několika dní či hodin na sekundy). Uživatelský vstup byl zadáván formou textových příkazů ze speciální sady příkazů. Výstup byl zobrazen ve formě textu. Jedná se tedy o *request-response* (požadavek-odpověď) komunikaci mezi uživatelem a terminálem. Díky zrychlené komunikaci bylo možné proces v reálném čase ovlivnit a získat lepší zpětnou

vazbu v případě chyby. Stále však byla potřeba znalost příkazů a interakce s počítačem nebyla intuitivní.
Příkazový řádek, jako uživatelské rozhraní, je dodnes hojně využíván, především pak pro svoji rychlost a efektivitu.

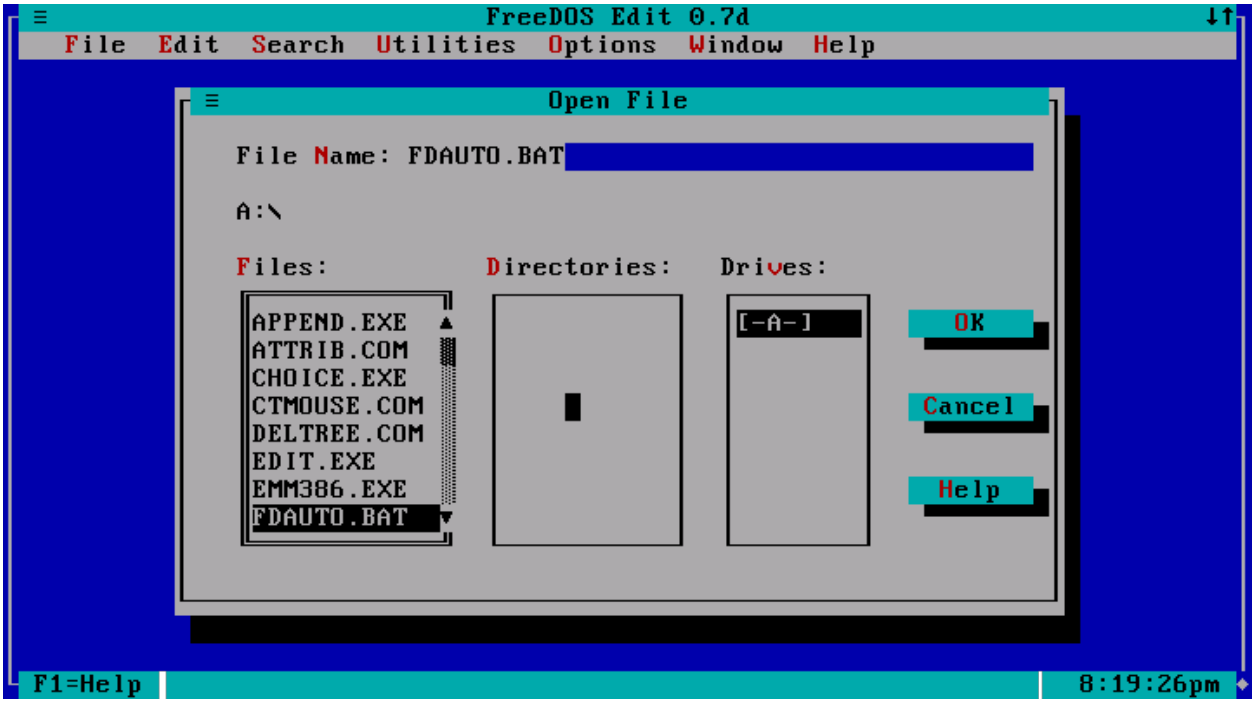
Textové uživatelské rozhraní

Textové uživatelské rozhraní neboli *Text user interface (TUI)* je vývojový mezistupeň mezi rozhraním příkazové řádky a *grafickým uživatelským rozhraním (GUI)*. Textové rozhraní, jak už název napovídá, pracuje v textovém režimu při kterém je obrazovka rozdělena na dvojrozměrný rastr. Každá pozice rastru může zobrazovat pouze jeden znak (nejčastěji znaky ASCII tabulky). Z těchto znaků je potom složeno celkové uživatelské rozhraní.
Revoluční posun nastal s příchodem myši a klávesnice pro ovládání celo-obrazové aplikace. Uživatelé mohli využívat prvky, které známe z novodobých GUI prostředí – tlačítka, menu, posuvníky, rolovací seznamy a tak dále. Programy se staly uživatelsky přívětivější a poskytovaly rychlejší a srozumitelnější zpětnou vazbu.

Obrázek 4: Dostupné znaky na tvorbu uživatelského rozhraní.

Obrázek 5: IBM Common User Access (CUA).

Systems Application Architecture (SAA) bylo uvedeno v roce 1987, jedná se o skupinu standardů počítačového softwaru vytvořených firmou IBM. Hlavním záměrem bylo unifikovat styl fungování počítačového softwaru. Rodina standardů SAA obsahuje standard pro uživatelské rozhraní operačních systému a počítačových programů *Common User Access (CUA)*.



Obrázek 6: Textové uživatelské rozhraní – FreeDOS.

Tento standard zavedl uspořádání ovládacích prvků tak, jak jej známe dnes[2]. Definoval například pozici status baru (více v sekci [17]), drop-down menu (více v sekci [17]) a konzistenci klávesových zkratek napříč softwarem.

[2] Tento standard silně ovlivnil počátky operačního systému Windows, částečně je rovněž použit ve variantách operačního systému UNIX.

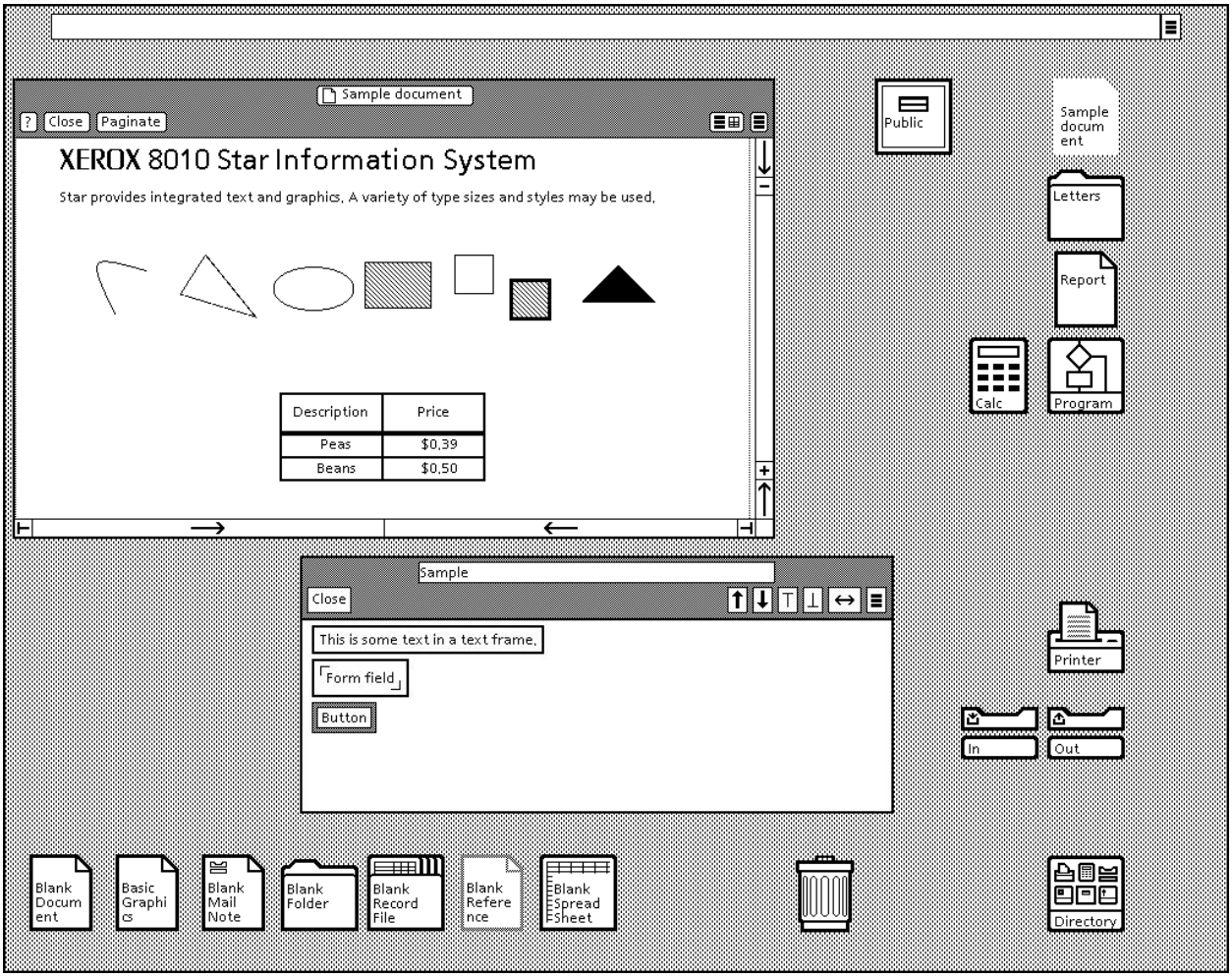
Grafické uživatelská rozhraní

V roce 1968 přichází Douglas Engelbart s demonstrací systému *oN-Line System (NLS)*, který využíval myši, ukazatele, hypertextu a několika oken. Dále systém umožňoval video konference, kontrolu revizí a kolaborativní editor v reálném čase. Je však nutno podotknout, že se jednalo o prototyp a na podobné, masově rozšířené zařízení, si museli uživatelé ještě nějakou dobu počkat.

Obrázek 7: První prototyp počítačové myši.

WIMP paradigma

Windows, Icons, Menus, Pointers (WIMP je paradigma vytvořené výzkumníky z Xerox Palo Alto research center (Xerox PARC), který je často zaměňován s pojmem GUI. WIMP je součástí obecného GUI, není však nutné aby obecné GUI využívalo prvky WIMP paradigma. V dnešní době je WIMP paradigma stále hojně využíváno, jak ale uvidíme později, je postupně nahrazováno *Post-WIMP paradigmatem*.



Obrázek 8: WIMP paradigma.

Xerox Alto

V roce 1973 byl představen *Xerox Alto*, první počítač navržený s důrazem na GUI[3]. Jednalo se o počítač určený pro kancelářskou práci.

[3] Založeno na principu NLS

Pro vstup byla použita klávesnice a myš. Rozhraní jednotlivých aplikací a nástrojů nebylo zcela jednotné, ale umožňovalo používat více oken najednou. Kvůli jeho vysoké ceně však nebyl komerčně úspěšný[4]. I přes jeho neúspěch byl Xerox Alto vnímán jako budoucnost uživatelského rozhraní. Roku 1979 Steve Jobs sjednal pro svůj tým inženýrů ukázku použité technologie v Xeroxu výměnou za možnost nákupu akcií firmy Apple.

[4] Zde vidíme, že i v případě dobrého designu je často složité vyrobit komerčně úspěšné produkty. Cena Xerox Alto byla po přepočtu (rok 2017) cca 108 075 USD.

Atari 2600

Nejpopulárnější herní konzole z takzvané druhé generace herních konzolí spatřila světlo světa v roce 1977. Jednalo se o jednu z prvních konzolí, které využívaly výměnné moduly s hrami. Ke konzoli bylo možné používat tří typy ovladačů: 8 směrový joystick s jedním tlačítkem, paddle (analogový ovladač s jednou osou) a číselné klávesnice s 12 klávesami.

Obrázek 9: Atari 2600.

Xerox Star

V roce 1981 se Xeroxu povedlo komercializovat koncept Xerox Alto ve formě produktu s názvem *Xerox Star* – značně modifikované

verze počítače Xerox Alto. Jednalo se o první komerčně úspěšný počítač, který využíval dodnes používané koncepty – bitmapový display, grafické prostředí založené na oknech, ikony, složky, myš s dvěma tlačítky, ethernet připojení, souborový server, tiskový server a e-mail. Rovněž se poprvé objevuje metafora pracovní plochy a ikon. V tomto přístupu hrálo klíčovou roli napodobení kancelářských předmětů se snahou vytvořit intuitivní uživatelské prostředí. Takový přístup k vývoji uživatelského rozhraní nazýváme *objektový*. Textový procesor obsahoval objekty jako odstavec, stránka, věta a další. Uživateli bylo umožněno s objekty interagovat a provádět různé operace. Koncept *What you see is what you get (WYSIWYG)* (česky „co vidíš, to dostaneš“) byl pro Xerox Alto klíčový.

Obrázek 10: Xerox Alto.

Apple Lisa a Macintosh

Po návštěvě Xerox PARC tým Apple okamžitě přesměroval projekt *Apple Lisa* z textového uživatelského rozhraní na grafické[5].
Ve stejné době byl spuštěn projekt *Macintosh*, u kterého byl kladen důraz na nízkou pořizovací cenu a přívětivost pro běžného uživatele[6]. Originální koncept obsahoval textové uživatelské prostředí, které umožňovalo přepínat aplikace pomoci klávesových zkratek. Během vývoje hardware bylo docíleno kombinace nízké ceny počítače Apple II a výpočetního výkonu Apple Lisa. Tato souhra umožnila pozdější nasazení grafického uživatelského rozhraní podobného tomu na Apple Lisa. Tímto si projekt získal pozornost Steva Jobse, který okamžitě uviděl skrytý potenciál a v roce 1984 se Macintosh dostává do prodeje.

[5] Apple Lisa měl být nástupcem Apple II, který používal textové uživatelské rozhraní. Apple Lisa měl původně v tomto trendu pokračovat.

[6] Během této doby můžeme pozorovat souboj firem o vytvoření prvního uživatelsky přívětivého počítače

Microsoft Windows 1.0 až 3.1

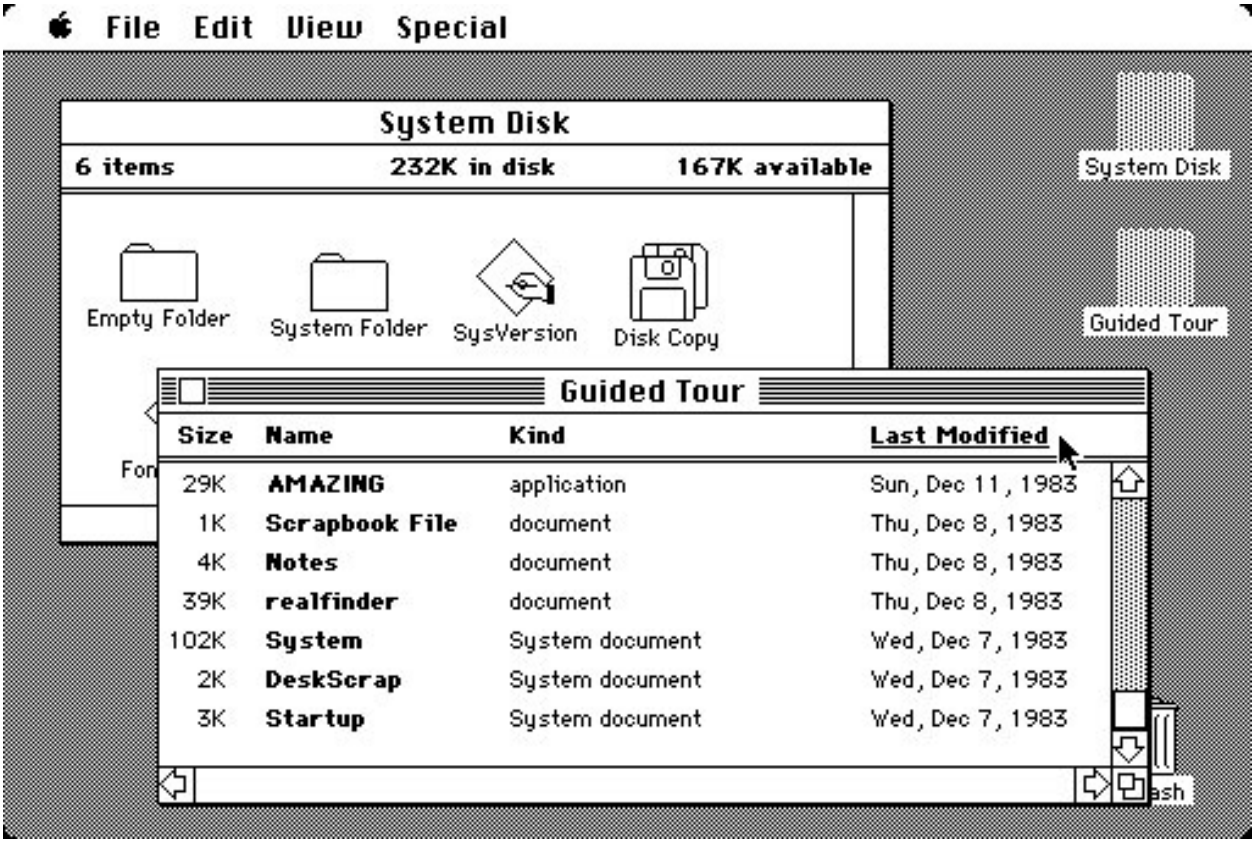
V roce 1985 začal Microsoft pracovat na programu s názvem *Interface Manager* (později oznámen pod názvem *Windows*).

Obrázek 11: První Apple Macintosh 128k.

Windows 1.0 Navržen s cílem soupeřit s operačním systémem firmy Apple, nezískal však velkou popularitu. Nejednalo se o samostatný operační systém, ale o nástavbu operačního systému *MS-DOS*. Windows 1.0 nepodporoval překrývání oken (dostupné byly pouze modální dialogové boxy – o těch se dozvíme více později[17]) a neobsahoval pracovní plochu.

Windows 2.0 Již podporoval překrývání oken. Tato změna způsobila žalobu ze strany firmy Apple.

Windows 3.1 Byl první populární operační systém od firmy Microsoft. Stále však neobsahoval pracovní plochu.



Obrázek 12: Macintosh UI.

Microsoft Windows 95

V roce 1995 vzniká operační systém *Windows 95*, který již obsahoval uživatelské rozhraní založené na objektech a metaforách. Byla zavedena ikonická *nabídka Start*, *TaskBar* (panel na spodní hraně obrazovky), pracovní plocha a koš. Tyto prvky uživatelského rozhraní vidíme v operačních systémech Microsoft Windows dodnes.

Nokia 2110

V roce 1994 firma Nokia představila mobilní telefon *Nokia 2110*. Telefon byl znatelně menší než konkurenční zařízení a obsahoval větší display. Díky těmto parametrům se stal jedním z nejoblíbenějších telefonů své doby. Z hlediska uživatelského rozhraní přinesl koncept dvou dynamických tlačítek, kterým se kontextově měnila přiřazená funkcionalita. Na základě těchto tlačítek přišla později firma Nokia s konceptem již dobře známého *Navi-Key* u modelu *Nokia 3110*.

Obrázek 15: Nokia 2110.

Obrázek 13: Reklama na Windows 1.0.

Palm (PDA)

Zařízení od firmy Palm jsou často vnímána jako první masově rozšířené mobilní počítače. V roce 1996 firma produkuje první *PDA PalmPilot 1000* poháněné operačním systémem Palm OS. Zařízení obsahovala monochromatický display s rozlišením 160×160px. Ve spodní části obrazovky byla takzvaná *Graffiti input zone*. Speciální zóna rozpoznávala jednotlivé znaky, které mohl uživatel za pomoci speciálních značek zapisovat. Tyto značky byly navržené tak, aby je bylo možné napsat jedním tahem.

Post-WIMP paradigma

Postupem času začalo být WIMP paradigma rozšiřováno o další principy. Dělo se tak především z důvodů nástupu většího nasazení

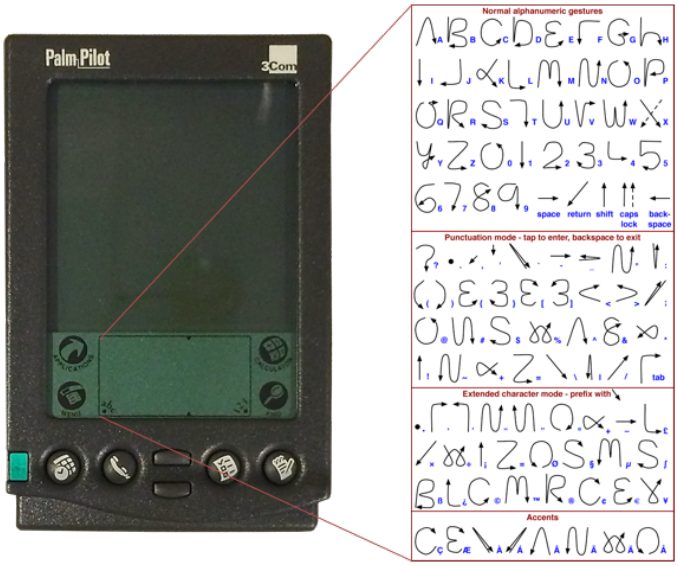
Obrázek 14: Windows 95 vs. Windows 10.

CAD systémů, interaktivních video her a práce s větším množstvím dat. WIMP paradigma vyžadovalo použití nemalé části obrazovky pro zobrazení uživatelského rozhraní, v mnoha aplikacích začalo být výhodnější přicházet s vlastním řešením.
Většina běžných počítačů však stále funguje na principu WIMP paradigma. Velký posun nastal s nástupem zařízení využívající dotyk jako uživatelský vstup, webových aplikací a virtuální reality.

Apple iPhone

Před vydáním prvního telefonu *Apple iPhone* na trhu dominovaly firmy jako Nokia a Motorola. Tyto firmy těžily primárně z potřeb korporátních zákazníků a prodávaly nespočet modelů určených různým cílovým skupinám. V roce 2007 Steve Jobs představil první Apple iPhone, telefon, který se pokusil sjednotit vše do jednoho zařízení.
Průlomové bylo použití kapacitního dotykového displaye. Pro ovládání telefonu nebylo nutné používat přesný hrot stylusu. Stačilo se dotknout obrazovky v místě zobrazeného tlačítka. Telefon stále ob-

Obrázek 17: Apple iPhone první generace vs. iPhone Xs.

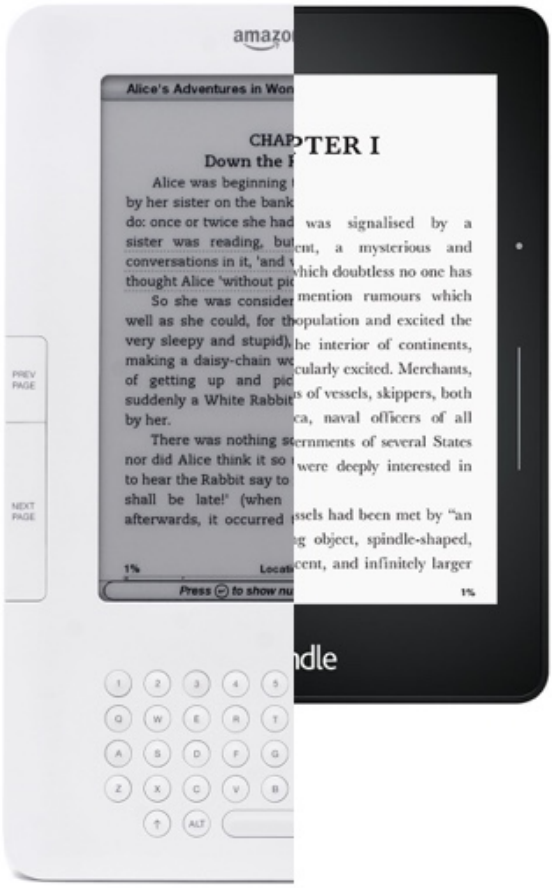


Obrázek 16: PalmPilot 1000 a Graffiti input zone.

sahoval i hardwarová tlačítka. Hlavní, takzvaný *Home button*, sloužilo k rychlému návratu na hlavní obrazovku. Ostatní pak umožnila změnu hlasitosti a uzamčení telefonu. Zbytek funkcí bylo ovládáno dotykem na základě kontextu aktuálně používané aplikace. V roce 2007 se jednalo o nevídaný průlom.
Apple iPhone poháněl operační systém iOS (dříve označován jako OS X – s Mac systémem OS X sdílel UNIXový základ). Tento operační systém v době spuštění nepodporoval externí aplikace ani funkci *copy&paste* – tyto funkce byly přidány později.

Amazon Kindle

Myšlenka elektronického papíru se objevila již v roce 1997. Společnost E Ink Corporation si dává za cíl vytvořit zobrazovací obrazovku která nepotřebuje umělé podsvícení.
V roce 2007 firma Amazon na trh uvedla první generaci *Amazon Kindle* – nejpopulárnější čtečku elektronických knih. U prvních generací Amazon Kindle bylo ovládání zprostředkováno pomoci klávesnice a tlačítek, později se objevují verze s kapacitní vrstvou, které umožňují podobný styl ovládání jako dotykové telefony.
Elektronický papír však přináší mnohá omezení. Jedná se především o pomalou rychlost překreslení a absenci barev. Tyto limitace bylo nutné při vytváření uživatelského rozhraní zohlednit.



Obrázek 18: Amazon kindle (2. gen) vs. Amazon Kindle Paperwhite 4.

Apple iPad

„Strategy is really simple. What we want to do is we want to put an incredibly great computer in a book that you can carry around with you and learn how to use in 20 minutes ... and we really want to do it with a radio link in it so you don't have to hook up to anything and you're in communication with all of these larger databases and other computers."

Steve Jobs, 1983

V roce 2010 přichází Apple s dalším revolučním produktem – *Apple iPad*. Apple iPad byl narozdíl od Apple iPhonu větší a výkonnější, poukázal tím na možnou budoucnost osobních počítačů. Stejně jako Apple iPhone, Apple iPad obsahoval kapacitní dotykovou vrstvu – je tedy umožněno ovládání dotykem. Během následujících let se tě-

šil Apple iPad mimořádné oblibě mezi zákazníky. V roce 2015 Apple vydal *iPad Pro* s podporou „nového“ typu uživatelského vstupu – *Apple Pencil*. Jedná se o chytrý stylus, který umožňuje ergonomické psaní a kreslení. Firma Apple se za tento krok stala terčem vtipů a kritiky.
Prozatímní vývoj naznačuje, že v blízké budoucnosti většina běžných uživatelů nahradí osobní počítače právě tablety.
V roce 2019 Apple oznámil operační systém iPadOS, který na iPadu nahradil operační systém iOS. Operační systém iPadOS přinesl důležité funkce jako je podpora práce s externím diskem, plnohodnotný webový prohlížeč, podporu externí myši a mnoho dalšího. Tento krok ještě více potvrzuje směr budoucnosti osobních počítačů.

Obrázek 19: Apple iPad první generace vs. iPad Pro 2018.

Kinect

Zařízení podobné webové kameře vyvíjené společností Microsoft pro herní konzole *Xbox 360/One*. Přineslo ovládání her pohybem celého těla případně hlasovými příkazy.

Natural-language user interface (NLUI)

Využívá rozpoznání přirozené řeči ke komunikaci mezi uživatelem a počítačem. Popularita tohoto uživatelského rozhraní vzrostla s příchodem osobních asistentů Apple Siri (2011), Google Assistant (2016), Amazon Alexa (2014) a dalších.
Uživatelské rozhraní založené na rozpoznávání přirozeného jazyka má velký potenciál v jednoduchosti ovládání, zatím je však problém správně a rychle porozumět příkazům od uživatele. Další limitací je absence méně rozšířených jazyků (například češtinu). I přes problémy má tento přístup velký potenciál, a to především mezi běžnými uživateli osobních počítačů a domácích asistentů. Především v anglicky mluvících zemích se staly tyto produkty velmi populární.

Chytré hodinky

Především kvůli jejich menším rozměrů bylo nutné věnovat větší pozornost uživatelskému rozhraní. Obrazovka je natolik malá, že dotyk prstu většinu plochy překryje. Především pro navigaci je tedy nutné využívat dalších ovládacích prvků. Rovněž bylo nutné pracovat s již existujícím mentálním modelem (o mentálních modelech si více řekneme později[9]). Příkladem je hodinová korunka původně sloužící k nastavování času[7]. U chytrých hodinek se s korunkou pracuje podobným způsobem, slouží však k navigaci či jiným účelům.

[7] Dříve navíc sloužící k natahování pérka hodinového strojku

Oculus Rift

V roce 2012 vzniká projekt s názvem Oculus Rift – brýle a sada ovladačů sloužící ke zprostředkování *virtuální reality (VR)*. Brýle obsahují pohybové senzory, sluchátka a dvojici obrazovek. V roce 2014 firmu stojící za Oculus Rift kupuje firma Facebook.
Dnes patří virtuální realita k běžně dostupným rozšířením pro herní konzole a osobní počítače. Komerčně dostupnější a úspěšnější se jeví *rozšířená realita*, která umožňuje sledovat okolí skrze obrazovku mobilního zařízení nebo speciálních brýlí. Tato zařízení určitým způsobem modifikuje pozorované okolí.

Playstation 4 - DualShock 4

Sony Interactive Entertainment přichází v roce 2013 se čtvrtou generací herní konzole *Playstation*. Z uživatelského hlediska nás nejvíce zajímá herní ovladač – *DualShock 4*. Poprvé v historii Playstation konzole se objevuje dotykový panel, podobný dotykové plochy (touchpad) z notebooků. Dotykovou plochu mohou uživatelé využít například k psaní na virtuální klávesnici.

Obrázek 20: Ovladač DualShock 4.

Shrnutí

Ukázali jsme stručný přehled historie uživatelského rozhraní a uvědomili si hlavní zlomové body vývoje. Vzhledem k jeho rychlejsi je další vývoj možné pozorovat každý den kolem nás, zkuste se na to zaměřit.

Úkoly

Úkol 1
Na webové stránce `web.archive.org` si prohlédněte historii některého z populárních webů (např. `www.seznam.cz`).

Úkol 2
Pokuste se během jednoho dne soustředit na interakci s nástroji a softwarem který používáte, poznatky si zapisujte.

Design a psychologie

Proč by nás měl obecný design vůbec zajímat? Jaký dopad může mít špatný design? V následující kapitole se pokusíme na tyto otázky odpovědět. Kapitola je volným přepisem základních myšlenek Donalda A. Normana z knihy „Design of everyday things“ [11].
Obecný design se zabývá interakcí lidí s věcmi, tím jak věci fungují a jak jsou ovládány. Pokud je design uchopen správně, výsledkem je produkt, který se používá příjemně a intuitivně. V opačném případě se velice často dostavuje zklamání a frustrace.
Veškerá pravidla uvedená v následující kapitole jsou aplikovatelná na jakoukoli formu designu. Ať už se jedná o design uživatelského rozhraní, nebo fyzické věci – priority jsou stejné.
Zkušenost ukazuje, že je možné produkt navrhnout způsobem, který eliminuje uživatelovu frustraci. Dobrý design přináší vodítka, špatný opak.

Průvodce studiem
Principy uvedené v této kapitole patří mezi nejdůležitější pilíře návrhu uživatelského rozhraní, které jsou, narozdíl od konkrétních prvků uživatelského rozhraní, poměrně stabilní.

„Lidská mysl je mimořádně bohatě vybavená k tomu, aby zkoumala svět a objevovala v něm smysl a řád.“

Donald A. Norman,

Lidé z technologických odvětví se často snaží uchopit design technickou intuicí. Ta je však často limitována omezenou/naivní znalostí lidského chování. Pokud má být produkt použitelný pro všechny uživatele, je vhodné si tyto znalosti rozšířit. Z těchto důvodů se nejprve seznámíme s vybranými základy kognitivní psychologie, které tvoří pevný základ každého kvalitního designu.

Výzva designu ve vývoji softwaru

Správný design vyžaduje spolupráci několika disciplín. Je nedostačující aby byla aplikace pouze funkční a stabilní nebo pouze graficky přívětivá. V komerční sféře se často setkáme s názory, že je design příliš drahý, případně do něj firma nechce investovat tak velké prostředky jako do jiných částí vývoje, připomeneme však citát:

„If you think good design is expensive, you should look at the cost of bad design.“

Dr. Ralf Speth, Chief Executive Officer, Jaguar Land Rover

V kontextu vývoje softwaru je často složité vysvětlit důležitost správného designu. Jak už jsme zmínili výše, lidé s technickým vzděláním přirozeně preferují (technickou) funkcionalitu nad estetikou – tento jev je viditelný na většině open-source projektů.
Ze studie [4] vyplývá, že uživatelé jsou obecně tolerantnější pokud je design esteticky přívětivý. Nesmíme zapomínat, že cílem mnoha aplikací je lidem ulehčit a zpříjemnit život, mnoho produktů však dělá pravý opak.

Průvodce studiem
Lidé převážně z technických odvětví často zapomínají, že uživatelské rozhraní má sloužit především uživatelům – lidem.

„Users are more tolerant of minor usability issues when they find an interface visually appealing. This aesthetic-usability effect can mask UI problems and can prevent issue discovery during usability testing. Identify instances of the aesthetic-usability effect in your user research by watching what your users do, as well as listening to what they say.“

Kate Moran, Nielsen Norman Group

Human-centered design (HCD)

Přístup k designu, který prioritizuje požadavky a nároky uživatele nazýváme *Human-centered design (HCD)*. Správný design umožňuje (ať už neverbální nebo verbální) komunikaci uživatele s zařízením. Ta je důležitá především v momentech, kdy se věci nedaří jak mají a uživatel má problémy dosáhnout stanoveného cíle. Proto je nutné, aby se designeři soustředili více na situace, ve kterých nastane problém, než na situace, kdy jde vše hladce.
Uživatel používající nový software má většinou jeden hlavní cíl, zjistit jak s daným softwarem pracovat. To obnáší zjistit jaké funkce program obsahuje a jak se používají – tento proces je možný díky *viditelnosti*. Viditelnost vzniká kombinací šesti principů: *afordance*, *signifier*, *mapování*, *zpětná vazba*, *omezení*, a (nejpodstatnější) *mentální model*.

Afordance

Pojem který zavedl James J. Gibson vyjadřuje vztah mezi předmětem a organismem, nikoli však charakter samotného objektu. *Afordance* obsahuje možné způsoby interakce vyplývající z fyzické podoby předmětu.
Uveďme příklad – počítačová myš. Obsahuje dvě tlačítka, které evokují možnost stisku. Při pohybu myši obdržíme zpětnou vazbu

Obrázek 21: Konvička a šálky na čaj.

ve formě pohybujícího se kurzoru, pochopíme tedy, že je zamýšleno s myší pohybovat. Afordance již neřeší kam a kdy budeme klikat, případně jak rychle budeme s myší pohybovat. K tomuto účelu slouží signifier.

Úkol 3
Zkuste se podobným způsobem jako tomu bylo u počítačové myši zamyslet nad objekty na obrázku 21.

Signifier

Signifiery explicitně vyjadřují jaké akce jsou uživateli dostupné a jak mají být využity. Jedná se tedy o signály – nápisy, značky, ikony umístěné okolo nás. Dobrým příkladem je nápis „TAM“ na dveřích, ikonka diskety na tlačítku pro uložení souboru a další.
V designu uživatelského rozhraní je signifier díky své přímé komunikaci důležitější než afordance.

Ze studie [5] vyplývá, že v případě *flat designu* uživatelé strávili průměrně o 22 % více času studováním zobrazené webové stránky. Flat design, který je založen na zploštění grafických elementů (odstranění stínů, barevných gradientů a dalších prvků) odstraňuje počet jednoznačných signifierů, které ulehčují identifikaci tlačítek a jiných důležitých ovládacích prvků.
Z průzkumu vyplynulo, že použití stínů nebo barevného přechodu ve výplni tlačítka situaci výrazně zlepšuje (obrázek 22). V případech, kde uživatelé tlačítko již očekávají, nemá rozdílný vzhled takový vliv.

Úkol 4
Zkuste se během dne soustředit na otvírání dveří, pozorujte zda jste byli nuceni číst nápis o směru otvírání.

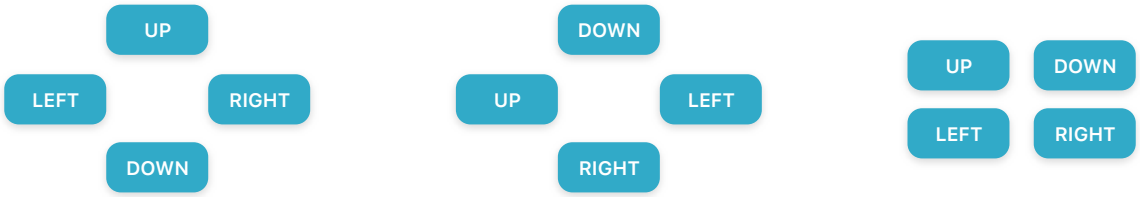
Obrázek 22: Demonstrace signifieru na případu tlačítka.

Signifier rozdělujeme na *účelový* (úmyslně vytvořený designerem) a *neúčelový* (samovolně vznikající). V některých případech je neúčelový signifier neškodný (nebo dokonce prospěšný).
Jako příklad můžeme uvést záložku knihy, její hlavní účel je označit místo ve kterém čtenář knihu odložil. Vzniká zde však i neúčelový signifier – uživatel může jednoduše sledovat pokrok ve čtení knihy. Extrémním příkladem by mohlo být označení tlačítka „smazat“ zelenou barvou a ikonou diskety. V tomto případě je pro uživatele nemožné odhadnout funkci tlačítka správně a bude nucen funkci nejprve vyzkoušet.

Mapování

Jedná se o vztah mezi objekty dvou množin. Představme si osvětlení na stropě, mapování mezi světly a vypínači určuje, který vypínač ovládá jaké světlo.
Mapovaní je z pohledu designu a uživatelského rozhraní důležitý koncept. Pojednává především o rozmístění ovládacích prvků, takzvaný *layout* (o něm si detailně povíme později). Pokud je rozmístění prvků správné, používání zařízení je intuitivní a jednoduché (připomeňme, že lidé jsou velice dobří v hledání souvislostí a jasných pravidel).
Takzvané *přirozené mapování* využívá reálných analogii a kulturních standardů. Příkladem zde může být rozmístění ovládacích prvku v liště nástrojů, od nejdůležitějších po méně důležité, a to zleva doprava. Toto rozmístění považujeme za intuitivní, protože jsme zvyklí číst zleva doprava. Jak víme tento směr čtení textu není jediným možným směrem, nejedná se tedy o všeobecně platnou pravdu.

Úkol 5
Pokud máte možnost, vyzkoušejte mapování vypínačů na stropní osvětlení.



Obrázek 23: Demonstrace správného a špatného mapování.

Úkol 6
Příklady správného a špatného mapování jsou demonstrovány na obrázku 23. Poznáte který z nich je nejlepší a proč?

Zpětná vazba

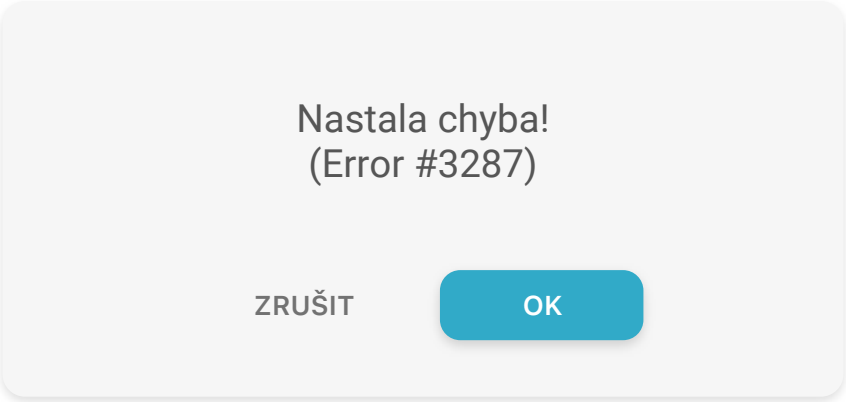
Zpětná vazba je komunikací výsledků akce. Kdykoli se snažíme na dotykovém telefonu stisknout tlačítko a nevíme zda jsme jej úspěšně aktivovali, je na vině špatná zpětná vazba. V takovém momentu je důležité, aby nás aplikace upozornila, že spuštěná operace probíhá a máme čekat na výsledek[8].

[8] V opačném případě víme, že aktivace nebyla úspěšná.

Lidem je mechanizmus zpětné vazby velice dobře znám. Bez nervového systému a jeho zpětné vazby bychom nebyly schopni uchopit žádný předmět. Z tohoto důvodu je nutné tyto mechanizmy přenést do světa virtuálního.
Důležité je vědět, že chybovat je lidské. Je-li někde možné udělat chybu, některý uživatel ji udělá. Designer musí s tímto faktem počítat a podat uživateli pomocnou ruku.

Rychlost zpětné vazby U zpětné vazby je podstatná její rychlost. V případě stisku tlačítka stačí zpomalení zpětné vazby o několik sekund a netrpělivý uživatel již stihne tlačítko stisknout několikrát.

Informativnost zpětné vazby Dalším podstatným aspektem zpětné vazby je její informativnost. Změna barvy tlačítka nebo zobrazení čísla chybového hlášení dá uživateli najevo, že se něco stalo, neví však co a proč. Taková zpětná vazba je lepší než žádná, nejedná se však o uspokojivé řešení. Příklad nedostatečné zpětné vazby je možné vidět na obrázku 24.



Obrázek 24: Příklad nedostatečné zpětné vazby.

Frekvence zpětné vazby Zpětná vazba by rovněž neměla být příliš častá. Uživatel v takovém případě ztrácí koncentraci na jeho cíl a jeho pokus končí frustrací. V horším případě začne varování ignorovat úplně, což v extrémních případech může vést ke katastrofě.

Kritické signály musí být prezentovány prioritně a důrazně, aby nedošlo k jejich ignoraci nebo přehlédnutí (například ukazatel baterie na obrázku 25).

Obrázek 25: Příklad s ukazatelem baterie.

Omezení

Většina věcí v našem okolí slouží na omezený počet úkonů.[9] Jestliže má objekt nebo aplikace vhodně zvolená omezení, usnadňuje to uživateli její použití. Uvádíme čtyři druhy omezení: *fyzická*, *kulturní*, *sémantická* a *logická*.

[9] Rovněž platí pravidlo, že čím méně je věc univerzální, tím lépe nabízené úkony zvládne.

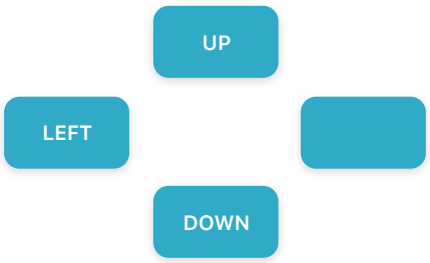
Fyzická omezení Limitují (fyzicky) možné operace. Do menšího otvoru na nůžkách nelze vložit více prstů, počet způsobů jak nůžky používat je tedy limitován. Baterie jsou navrženy tak, aby většina zařízení nešla přepólovat nesprávným vložením baterie. V případě USB kabelu bylo fyzické omezení orientace spíše obtěžující, v další generaci USB-C bylo toto omezení odstraněno.

Průvodce studiem
Vývojáři mají často tendence nabídnou uživatelům maximální volnost, ne vždy je to však správné rozhodnutí. Vždy si důkladně promyslete, jaké možnosti uživateli nabídnete, méně je často více.

Kulturní omezení Určují množinu možných akcí v rámci sociální situace. Již zmíněným příkladem je čtení zleva doprava. Uživatelé tedy očekávají, že aplikace budou tento kulturní zvyk respektovat. Problémem kulturního omezení je jeho nestálost – kultura je v neustálém vývoji.

Sémantická omezení Sledují význam dané situace a redukují počet možných akcí. Pokud uživatel pracuje s textem, není nutné zobrazovat ovládací panel pro editaci obrázku. Stejně jako kulturní omezení jsou i sémantická nestálá a podléhají neustálém vývoji. V kontextu uživatelského rozhraní se můžeme zamyslet nad sémantikou barev a tvarů. Dříve byla tlačítka plastická, dnes jsou uživatelé schopni považovat za tlačítko pouhý barevný text.

Logická omezení Nejlépe pozorovatelná v přirozeném mapování. Mezi vypínači a světly na stropě je jasná logická souvislost. Příkladem ze světa uživatelského rozhraní je situace ve které uživatel zná celou množinu akcí, nezná však celou množinu ovládacích prvků (demonstrováno na obrázku 26). Pro uživatele zpravidla není těžké odvodit funkce zbylých ovládacích prvků.



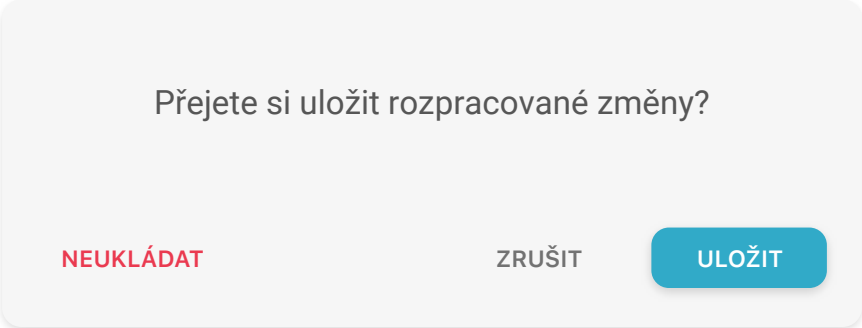
Obrázek 26: Demonstrace logického omezení.

Omezení můžeme rovněž rozdělit do následujících kategorii:

Interlocks Vynucení určitého pořadí operací. Příkladem může být mikrovlná trouba, kterou nelze otevřít bez jejího vypnutí. V kontextu uživatelského rozhraní se může jednat o vynucení vyplnění určitých údajů ve formuláři před jeho odesláním.
Interlock může mít i další formu – *pojistka mrtvého muže*. Strojvedoucí musí jednou za daný interval aktivovat tlačítko (pojistku) a potvrdit tím, že je plně schopen vozidlo řídit. V opačném případě se vlaková souprava zastaví. Toto využití omezení se týká převážně bezpečnosti.

Lock-ins Udržují akci v chodu, tak aby ji nemohl nikdo přerušit. Příkladem z uživatelského rozhraní je znemožnění ukončení programu bez uložení rozpracovaného projektu (obrázek 27).

Lockouts Opak lock-ins, zabraňují tomu, aby uživatel vstoupil kam nemá. Převážně používány pro zvýšení bezpečnosti. Příkladem můžou být dětské pojistky, pojistka na hasícím přístroji, vyžadování hesla pro žmněu důležitých nastavení a další.



Obrázek 27: Příklad využití lock-in.

Mentální (konceptuální) model

Mentální model je (často velice zjednodušené) vysvětlení toho, jak věci fungují. Pro uživatele není nutné, aby toto chápání bylo správné nebo přesné, stačí aby bylo funkční. Příkladem jsou soubory a složky v počítači. Víme, že se v počítači žádné fyzické složky ani soubory nenachází, i přesto je tento mentální model pro jeho používání podstatný.

„Intuition is nothing more and nothing less than recognition."
Herbert A. Simon, držitel Nobelovy ceny

Dalším příkladem je mentální model pro práci s e-mailovým klientem. Uživatel získá intuitivní představu, že veškeré přijaté a odeslané zprávy jsou uloženy na jeho počítači – všechny je tam totiž vidí. Uživatel s pokročilými znalostmi však ví, že je e-mailová komunikace uložena na serverech a do klientů se pouze synchronizuje.
Naivní mentální model je pro uživatele většinu času dostatečný. Nedostatky se dostaví v případě problému. Po výpadku internetu může být uživatel překvapen, že jeho půl roku stará pošta na počítači není dostupná.
Mentální modely umožňují uživatelům předvídat důsledky jejich akcí. Nejhorší situace nastane, pokud si uživatel na základě špatného designu vytvoří špatný mentální model.[10] Přirozeností člověka je snaha o hledání vysvětlení, proto nelze zabránit vzniku

[10] Klasická věta začínající slovy „To přece musí fungovat takto . . . "

mentálního modelu. Můžeme se pouze snažit o jeho co největší přesnost.
Příkladem může být termostat topení. Uživatel přijde domů a potřebuje co nejrychleji zatopit. Na základě zjednodušeného mentálního modelu nastaví teplotu na termostatu na maximální. Při bližším zkoumání funkce termostatu však zjistíme, že toto nastavení vytápění nijak neurychlí, naopak dojde s největší pravděpodobností na plýtvání energie. Základní termostat zná pouze dva stavy: topení zapnuto, topení vypnuto. Nastavení maximální teploty tedy neovlivní rychlost vytápění.

Myšlení rychlé a pomalé

V následující sekci založené na poznatcích ze stejnojmenné knihy [7] si krátce představíme dva hypotetické systémy v lidském mozku – *Systém 1* a *Systém 2*. Ačkoli se jedná o popis dvou oddělených systémů, tyto systémy nejsou fyziologicky rozděleny.

Průvodce studiem
Základním poznatkem plynoucí z této části je to, že uživatelé nebudou chtít přemýšlet a budou dělat ukvapená rozhodnutí. Vaši prioritou musí být tyto chyby elimitovat.

Systém 1 funguje automaticky a rychle, s malým nebo žádným úsilím a bez pocitu úmyslné kontroly.

Systém 2 přiděluje pozornost vědomým činnostem které ji vyžadují. Činnost Systému 2 bývá spojena se subjektivním prožitkem jednání, volby a soustředění.

Pokud provádíme již dobře známé akce, kontrolu převezme Systém 1 a naši pozornost můžeme věnovat ostatním úkonům. Proto je možné, že zkušený řidič zvládne řídit automobil a povídat si se spolujezdci – provádí dvě dobře známe akce a ani jedna z nich nevyžaduje jeho plně soustředěnou pozornost. Množství akcí které dokáže člověk v jeden moment provádět je ale omezeno. Pokud se řidič dostane do krizové situace, přestane okamžitě komunikovat a své plné soustředění věnuje krizovému brždění.
Systém 2 je na druhou stranu kritický pro učení se nových věcí. Když uživatel používá aplikaci poprvé, Systém 2 bude aktivnější. Jak moc tento proces bude pro uživatele náročný, záleží na kvalitě designu aplikace.
Systém 1 exceluje v rozpoznávání vzorů a aktivaci asociací, kterých si nejsme většinu času ani vědomi. Pokud tedy naše aplikace dodržuje obecné konvence, uživatel má možnost rozpoznat většinu ovládacích prvků intuitivně – za pomoci Systému 1[11]. Systém 1 je bohužel zodpovědný za většinu omylů, lidé mu totiž důvěřují více než Systému 2. Situace je vyhodnocena rychleji než Systém 2 stihne zasáhnout a situaci přehodnotit.

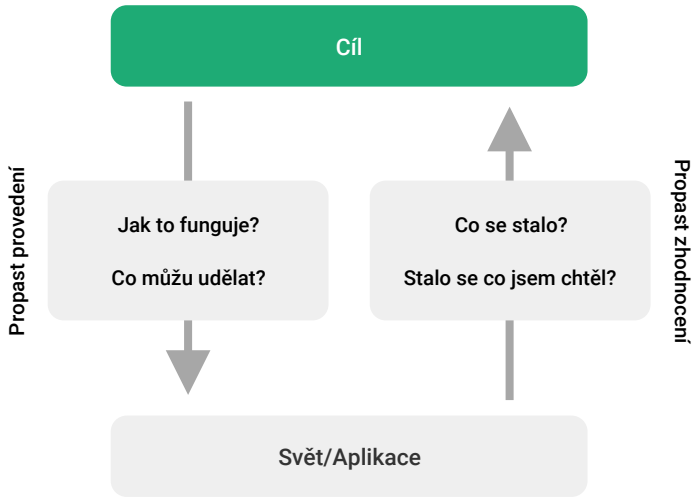
[11] Z tohoto důvodu je důležité neměnit význam ovládacích prvků napříč aplikacemi

Konceptuální model lidského chování

V této sekci se budeme zabývat interakcí lidí s technologiemi. Co se stane když věci nejdou podle plánu? Jakým způsobem rozpoznáme, že věci nefungují? Pro ilustraci tohoto procesu využijeme analogii dvou propastí: *propast provedení* a *propast zhodnocení* (obrázek 28).

Propast provedení Symbolizuje situaci ve kterém uživatel zjišťuje jak s daným objektem či aplikací pracovat. Skládá se z plánování, specifikace a provedení.

Propast zhodnocení Symbolizuje situaci, ve které uživatel vyhodnocuje stav objektu nebo aplikace. Skládá se z pozorování, vyhodnocení a porovnání.



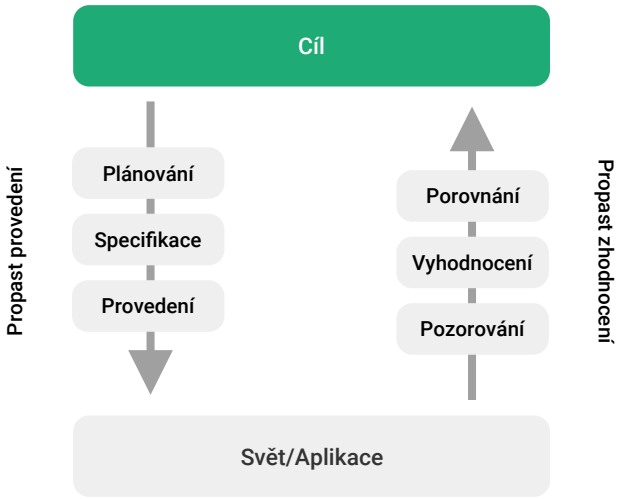
Obrázek 28: Analogie dvou propastí.

Velikost těchto propastí ovlivňuje kvalita designu. Propast provedení je zmenšena vhodným použitím signifierů, omezení, mapování a mentálního modelu. V případě propasti zhodnocení pomáhá zpětná vazba a kvalitní mentální model.

Sedm fází akce

K překonání výše zmíněných propastí slouží cyklus *Sedmi fází akce* (obrázek 29):

1. **Cíl** (formování cíle)
2. **Plánování** (akcí)
3. **Specifikace** (sekvence akcí)
4. **Provedení** (sekvence akcí)
5. **Pozorování** (stavu světa/aplikace)
6. **Vyhodnocení** (vnímání)
7. **Porovnání** (výstupu s cílem)



Obrázek 29: Sedm fází akce.

Cyklus sedmí fází akce začíná stanovením nového cíle případně reakcí na externí podnět. To určí jakou propast musí uživatel překonat jako první.

Definicí cíle se zabývá takzvaná *root cause analysis* – definici cíle zkoumá opakovaným kladením otázky „Proč?“.

„People don't want to buy a quarter-inch drill, they want a quarter-inch hole.“

Theodore Levitt, ekonom

Výše zmíněný citát ilustruje často zapomínaný fakt. Uživatelé používají aplikace z konkrétního důvodu, chtějí dosáhnout konkrétního cíle. Proto je důležité, aby aplikace dokázaly těmto poměrně vzdáleným požadavkům vyhovět.

Úrovně lidského vnímání

Lidské vnímání lze popsat třívrstvým modelem obsahující následující vrstvy: *viscerální*, *behaviorální* a *reflektivní*.

Viscerální úroveň Je úrovní základní, někdy označována jako *ještěrčí mozek*. Pro lidi je tato úroveň jednotná. Obsahuje základní instinkty a ochranné mechanismy. Odezva je rychlá a automatická.
Z pohledu designu se na této úrovni projevuje okamžitá odezva uživatele, převážně ve formě kladného nebo negativního pocitu. Na této úrovni rozhoduje vzhled, dotek, přívětivost zvuku a další. Nejedná se o hodnocení použitelnosti.

Behaviorální úroveň Naučené dovednosti aktivované podvědomou reakcí na známou situaci.
Z pohledu designu je na této vrstvě nejdůležitější provázanost akcí s očekáváním, které je v cyklu zpětné vazby potvrzeno nebo vyvráceno. To vede ke spokojenosti nebo zklamání uživatele.

Reflektivní úroveň Vědomé uvažování. Jedná se o úroveň, kde vzniká hluboké porozumění, uvažování, vědomé rozhodování a predikce výsledků budoucích akcí. Reflektivní úroveň je v porovnání s viscerální a behaviorální pomalá.
Z pohledu designu se jedná o nejdůležitější úroveň. Vzpomínky, které vedou k pozitivnímu nebo negativnímu hodnocení aplikace, si uchováváme déle než impulzivní reakce.

Kvalitní design se projeví na všech výše zmíněných vrstvách. Rozšířené schéma dvou propastí můžeme vidět na obrázku 30.

Obrázek 30: Úrovně zpracování v kontextu semi fází akce.

Lidská paměť

V následující sekci si stručně představíme základní rozdělení lidské paměti a její vliv na design. Po vzoru knihy [11] rozdělíme lidskou paměť na kontext *znalostí v hlavě* a *znalostí ve světě*.

Průvodce studiem
Pokud to není nezbytně nutné, uživatelovi bychom neměli zbytečně zatěžovat paměť. Snižujeme tím prostor na chybu nebo negativní emoční reakci.

Znalosti v hlavě

Jsou to právě statické informace – telefonní čísla, hesla, data narození a další, které si lidé nejčastěji vybaví v kontextu lidské paměti. Tuto kategorii informací můžeme nazvat znalostmi v hlavě.

Typy lidské paměti

Paměť rozdělujeme na tři typy: *senzorickou*, *krátkodobou* a *dlouhodobou* [15].

Senzorická paměť Slouží k okamžitému zachycení informace z některého z lidských smyslů. Díky této informaci získáme krátký čas navíc ke zpracování vjemu. Senzorická paměť se vyznačuje schopností uchovat velký objem informací na velice krátkou dobu (řádově pár vteřin).

Krátkodobá paměť Obsah krátkodobé paměti je uchován delší dobu (cca 15–20s) než v případě senzorické paměti. Díky krátkodobé paměti jsme si schopni na uvedenou dobu zapamatovat například telefonní číslo. Dobu, kterou jsme schopni telefonní číslo „udržet v hlavě“ můžeme prodloužit opakováním. Krátkodobá paměť ukládá informace ve formě přirozeného jazyka. To znamená, že telefonní číslo v krátkodobé paměti uložíme ve formě číslic, které jsme se (nejenom) pro tento účel naučili.

Dlouhodobá paměť Slouží k dlouhodobému uložení informací a zakódování jejich pravého významu. Samotné uložení a získání informací trvá déle než tomu bylo u předchozích dvou typů paměti. Ukládání do dlouhodobé paměti probíhá jak záměrně (učení), tak samovolně (prožitek). Z praktického hlediska má dlouhodobá paměť neomezenou kapacitu, pro spolehlivé uchování znalostí je však dobré uvádět informace do vzájemných souvislostí. Podoba vzpomínek v dlouhodobé paměti není úplná. V paměti si uchováme pouhé střípky, z kterých poté mozek zpětně sestaví vzpomínku. Hlavním problémem dlouhodobé paměti je její organizace a cílené získání uložené informace. Vědci stále neví jak přesně je dlouhodobá paměť organizována.

Typ uchovávaných informací

Informace uložené v hlavě můžeme rovněž rozdělit na *svévolné* a *smysluplné*.

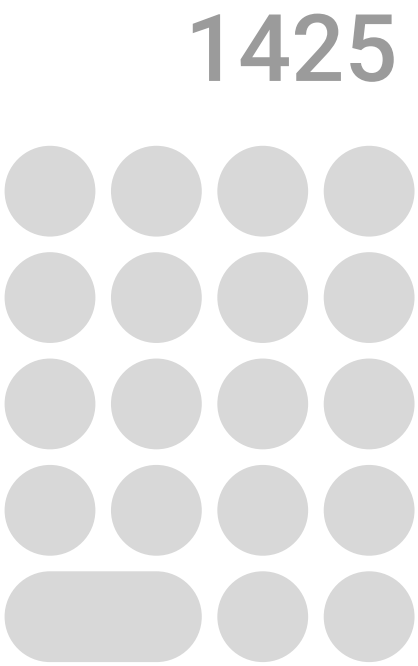
Svévolné informace Jedná se o mechanicky naučené informace bez hlubšího významu a vztahu k ostatním znalostem. Příkladem může být abeceda, jména lidí, klávesové zkratky a příkazy v příkazové řádce. Takové informace se nejlépe učíme opakováním tzv. *rote learning*. Jedná se však o učení pomalé a nespolehlivé, a to především v situacích, kdy chceme tyto znalosti použit pro řešení problému.

Smysluplné informace Informace založené na souvislostech. Většina věcí ve světě má smysl a strukturu, toho jsme schopni využít pro významně efektivnější učení. Dochází ke vzniku konceptuálního modelu, který nám slouží k pochopení a zapamatování jinak složitých systémů.

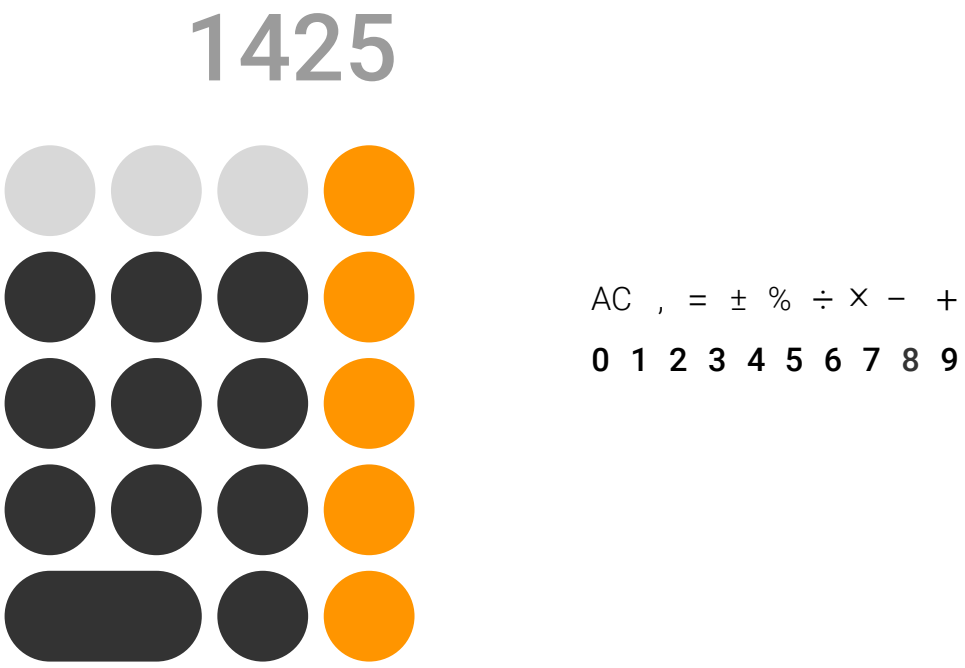
Znalosti ve světě

Častěji než si uvědomujeme používáme naše okolí pro snížení zátěže paměti. Pokud je znalost potřebná k provedení operace snadno dostupná z našeho okolí, není nutné si informaci pamatovat. V tomto kontextu se nejvíce projevují signifiers, omezení a přirozené mapování.
Situaci demonstrujeme na příkladu s kalkulačkou. Na obrázku 31 najdeme základní rozvržení kláves kalkulačky. V rámci úkolu 7 zkuste umístit popis na odpovídající klávesy. Barvy a výčet možných operací na obrázku 32 úkol značně zjednoduší – slouží jako znalosti ve světě, běžně totiž tyto informace v hlavě neukládáme, není k tomu důvod.

Úkol 7
Pokuste se umístit odpovídající popisy tlačítek na klávesnici kalkulačky, začněte obrázkem 31 a poté pokračujte obrázkem 32.



Obrázek 31: Základní rozvržení kalkulačky na telefonu.



Obrázek 32: Základní rozvržení kalkulačky na telefonu.

AC , = ± % ÷ × − +

0 1 2 3 4 5 6 7 8 9

Obrázek 33: Základní rozvržení kalkulačky na telefonu.

Millerovo magické číslo

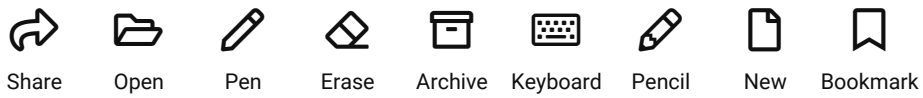
V roce 1994 George A. Miller publikuje v článku [10] poznatek o „magickém“ čísle sedm plus minus dva. Pojednává o schopnosti krátkodobé paměti udržet na dobu 20 sekund 7 (plus minus 2) pojmů. Dobu uchování těchto informací lze prodloužit mnemotechnickými pomůckami.

Z pohledu designu uživatelského rozhraní tedy nemůžeme očekávat, že si uživatel zapamatuje více než pět pojmů. Stejné pravidlo poté využijeme při seskupování objektů do skupin. Poté je pravděpodobnější, že si uživatelé zapamatují, kde se ovládací prvky nacházejí.

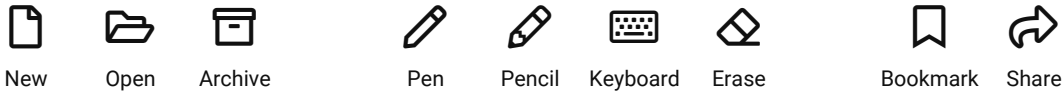
Pravidlo si ilustrujeme na obrázku nástrojové lišty (obrázek 34) na které nástroje nerozdělujeme do skupin. Pro uživatele je v tomto případě obtížné určit jaké ovládací prvky jsou k dispozici a kde přesně je najde. Na obrázku 35 jsou ovládací prvky rozděleny již správně – do menších logických skupin po třech až pěti prvcích.

Úkol 8
Zkuste si zapamatovat následující seznam věcí, poté alespoň 15 sekund seznam nečtěte a následně jej zkuste napsat na papír.

1. pes
2. auto
3. cukroví
4. počítač
5. plechovka
6. telefon
7. kniha
8. lahev
9. sluchátka
10. papoušek
11. lampa



Obrázek 34: Nesprávné pořadí prvků v ovládací liště.



Obrázek 35: Správné pořadí prvků v ovládací liště.

Zákony uživatelského rozhraní

Následující sekce je volným překladem výborného webu Laws of UX [16], na kterém nalezneme přehled většiny podstatných zákonů a poznatků z kognitivní psychologie, které jsou relevantní pro návrh uživatelského rozhraní. Některá z těchto pravidel jsme již zmínili dříve.

Efekt estetické použitelnosti

Klíčová myšlenka Esteticky příjemný design může učinit uživatele více tolerantním k drobným nedostatkům v použitelnosti. Esteticky příjemný design tedy může maskovat problémy s použitelností (a to i během testování).

Původ Tento jev byl poprvé zkoumán v odvětví human-computer interaction v roce 1995. Masaaki Kurosu a Kaori Kashimura z Hitachi Design Center testovali 26 variant UI bankomatů. Během testů objevili silnou korelaci mezi hodnocením estetického designu s dojmem uživatelsky přívětivého designu.

Doherty threshold

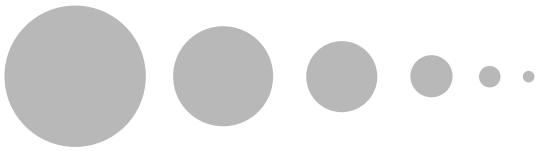
Klíčová myšlenka Produktivita se zvětšuje, když počítač a jeho uživatel reagují rychle (rychleji než 400 ms). Pokud tedy zajistíme rychlou odezvu systému, dokážeme udržet pozornost uživatele a zvýšit jeho produktivitu.

Původ V roce 1982 Walter J. Doherty a Ahrvind J. Thadani publikovali v IBM Systems Journal vědecký článek s poznatkem že ideální doba odezvy počítače je méně než 400 ms. Předchozí standard činil 2 000 ms. Pokud počítač zvládal reagovat na příkazy od uživatele do 400 ms, byl pro uživatele atraktivnější.

Fittsův zákon

Klíčová myšlenka Čas potřebný k zaměření cíle je funkce vzdálenosti a velikosti cíle. Objekty, které mají být jednoduše zaměřitelné by měly být větší a blíže uživateli. Toto pravidlo se vztahuje především na tlačítka a ostatní interaktivní prvky.

Původ V roce 1954 psycholog Paul Fitts zkoumal lidský motorický systém. Zjistil, že čas potřebný k přesunutí k cíli je závislý na vzdálenosti a velikosti objektu. Rychlý pohyb směrem k menšímu cíli způsobí větší chybovost.



Obrázek 36: Fittsův zákon.

Hickův zákon

Klíčová myšlenka Čas potřebný k učinění rozhodnutí se zvětšuje s počtem a komplexností volby. Počet možností je tedy vhodné redukovat. Toho můžeme dosáhnout zvýrazněním doporučených možností a dělením komplexních operací na více měnších kroků.

Původ Zákon je pojmenován po dvojci psychologů William Edmund Hick a Ray Hyman. V roce 1952 se rozhodli zkoumat vztah mezi počtem stimulů a délkou reakční doby. Jak můžeme očekávat, více stimulů vedlo k prodloužení reakční doby. Uživatelé, kteří jsou zahlceni možnostmi musí vynaložit více energie na jejich vyhodnocení a rozhodnutí.

Jakobův zákon

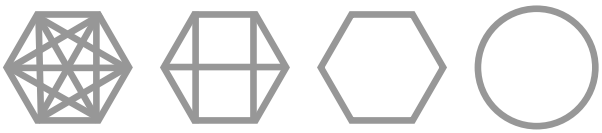
Klíčová myšlenka Uživatelé tráví většinu času na ostatních webových stránkách/programech. Nový produkt tedy bude srozumitelnější, pokud bude využívat stejné designové vzory.

Původ Autorem zákonu je Jakob Nielsen, který spolu s Dr. Donald A. Norman založili Nielsen Norman Group.

Zákon výstižnosti

Klíčová myšlenka Lidé mají tendenci vidět a interpretovat složitější tvary tím nejjednodušším možným způsobem. Lidské oko hledá jednoduchost a řád v komplexních tvarech předchází tak zahlcení informacemi. Tento přístup vyžaduje nejméně kognitivní námahy.

Původ V roce 1910 psycholog Max Wertheimer pozoroval řadu světel, které se rozsvítily na železničním přejezdu. Bylo to podobné světlům obklopující divadlo. Tím, že se světla postupně rozžínají a zhasínají pro pozorovatele vytváří efekt pohybujícího se světla.



Obrázek 37: Zákon výstižnosti.

Millerův zákon

Klíčová myšlenka Průměrný člověk je schopný zapamatovat si pouze sedum (plus minus dva) pojmů v jeho pracovní paměti. Shlukování je tedy efektivní metoda k prezentování skupin objektů. Obsah je dobré rozdělit do skupin po pěti až devíti prvcích.

Původ V roce 1994 George A. Miller publikoval [10] poznatek o „magickém“ čísle sedm plus minus dva. Pojednává o schopnosti krátkodobé paměti udržet na dobu cca 20 sekund sedum (plus minus dva pojmy). V praxi se ukazuje nejvhodnější rozmezí od tří do pěti pojmů. Dobu uchování těchto pojmů lze mnemotechnickými pomůckami prodloužit.

Occamova břitva

Klíčová myšlenka Pokud pro nějaký jev existuje několik vysvětlení, je dobré upřednostňovat to nejméně komplikované. V designu tedy analyzujeme každou jeho součást a zvážíme zda je nutné ji zachovat.

Původ Princip logické úspornosti má původ v 19. století od františkána Williama z Ockhamu.

Paretovo pravidlo

Klíčová myšlenka Princip Pareta tvrdí, že u mnoha událostí přibližně 80 % efektu pochází z 20 % příčin. Je tedy vhodné věnovat největší úsilí oblastem, které přinesou největší benefit pro většinu uživatelů.

Původ Ekonom Vilfredo Pareto vypozoroval, že 80 % italské půdy je vlastněno 20 % italské populace. Princip poměru 80/20 může po-

skytnout věrohodnou analýzu složitých systémů, včetně strategie pro tvorbu UI.

Parkinsonův zákon

Klíčová myšlenka Jakýkoli úkol bude prováděn tak dlouho, dokud nebude vyčerpán veškerý dostupný čas.

Původ Zákon formulován Cyril Northcote Parkinsonem jako část první věty v humorné eseji publikované v The Economist roku 1955. Později byla znovu otištěna v rámci knihy Parkinson's Law: The Pursuit of Progress (London, John Murray, 1958). Tento zákon odvodil z bohatých zkušeností v britské státní správě.

Postelův zákon

Klíčová myšlenka Buďme liberální v tom, co přijímáme a konzervativní v tom, co posíláme. Je důležité abychom byli empatičtí, flexibilní a tolerantní vůči jakémukoli množství akci, které může uživatel udělat. To obnáší práce se vstupem od uživatele, jeho validaci, definování omezení a poskytování zpětné vazby.

Původ Postelův zákon formuloval Jon Postel, časný průkopník Internetu. Zákon je návrhovou směrnicí pro software týkající se TCP a sítí.

Efekt pozice v posloupnosti

Klíčová myšlenka Uživatelé si nejlépe pamatují první a poslední položky v posloupnosti. Je tedy dobré umisťovat nejméně podstatné prvky do středu posloupnosti. Umístění klíčových prvků navigace na levou/pravou stranu může získat uživatelovu pozornost.

Původ Pojem který zavedl Herman Ebbinghaus popisuje jak pozice objektu v posloupnosti ovlivňuje schopnost zapamatování objektu.

Teslerův zákon

Klíčová myšlenka Teslerův zákon, známý také jako zákon o zachování složitosti, uvádí, že u každého systému existuje určitá složitost, kterou nelze redukovat.

Původ Během práce pro Xerox PARC roku 1980 si Larry Tesler uvědomil, že způsob, kterým uživatelé interagují s aplikacemi je stejně důležitý jako aplikace sama. Je výhodnější aby inženýr trávil

Obrázek 38: Efekt pozice v sekvenci.

měsíc času vývoje navíc, než aby tisíce uživatelů trávili o minutu déle času při používání produktu.

Von Restorff efekt

Klíčová myšlenka Pokud se vyskytuje několik podobných objektů, tak ten, který se nejvíce odlišuje od ostatních bude nejsnáze zapamatován. Z tohoto důvodu je dobré odlišovat klíčové akce od ostatních.

Původ Teorii vytvořila psychiatrička Hedwig von Restorff. Pozorováním zjistila, že si lidé nejlépe zapamatují nejvíce odlišný prvek z posloupnosti jinak podobných objektů.

Obrázek 39: Von Restorff efekt.

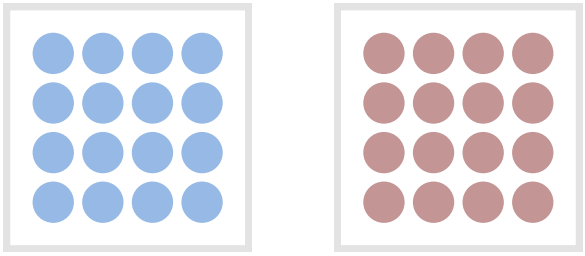
Zeigarnik efekt

Klíčová myšlenka Lidé si lépe pamatují nedokončený nebo přerušený úkol, než úkol dokončený. Je tedy důležité používat prvky jako progress bar (o tomto prvku si povíme více později) pro indikaci postupu uživatele.

Původ Bluma Wulfovna Zeigarnik byla psycholožka a psychiatrička. Objevila Zeigarnik efekt a přispěla k založení experimentální psychopatologie jako oddělené disciplíny po druhé světové válce.

Gestaltismus – zákon seskupení

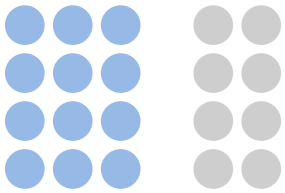
Klíčová myšlenka Objekty jsou vnímány jako jedna skupina, pokud se nacházejí v oblasti s jasně definovaným okrajem. Přidání rámečku (nebo viditelné mezery) okolo objektu nebo skupiny objektů je jednoduchý způsob jak objekty sjednotit.



Obrázek 40: Gestaltismus – zákon seskupení.

Gestaltismus – zákon blízkosti

Klíčová myšlenka Objekty které jsou blízko sebe jsou vnímány jako jedna skupina. Toto pravidlo umožňuje uživatelům seskupovat objekty do skupin na základě jejich vzájemné pozice.

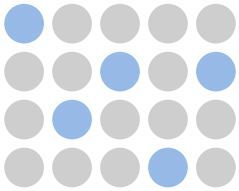


Obrázek 41: Gestaltismus – zákon blízkosti.

Gestaltismus – zákon podobnosti

Klíčová myšlenka Lidské oko má tendenci pozorovat podobné prvky v designu jako komplexní obraz, tvar nebo skupinu. A to i v případě, že jsou prvky odděleny mezerou. Z toho důvodu je

nutné vizuálně oddělit prvky, které mají uživatelé považovat za rozdílné. Například tedy navigaci od běžného textu.



Obrázek 42: Gestaltismus – zákon podobnosti.

Gestaltismus – zákon o vizuální spojitosti

Klíčová myšlenka Prvky, které jsou vizuálně podobné jsou vnímány jako související. Funkce podobného zaměření by tedy měly být vizuálně sjednoceny.

Původ Gestaltismu Gestaltismus (tvarová psychologie) tvrdí, že lidé přirozeně vidí objekty organizované do skupin či vzorů na základě abstrakce a konstrukce. To se děje protože lidský mozek rozpoznává skupiny a vzory na základě určitých pravidel a souvislostí.

Shrnutí

V této kapitole jsme si ukázali důležitost psychologie v oblasti tvorby uživatelského rozhraní. Dozvěděli jsme se, že lidské myšleni operuje ve dvou režimech a pro dosažení cíle používá kolečko sedmi fází akce. Dále jsme získali přehled o lidské paměti a nejdůležitějších zákonech, které můžeme využít při návrhu nejenom uživatelského rozhraní.

Kontrolní otázky
Odpovězte na následující otázky:

1. Jsou uživatelé tolerantnější pokud je aplikace esteticky přívětivější?
2. Jaký problém může nastat pokud bude uživatel zahlcen množstvím zpětné vazby?
3. Jaký ze systémů používáme při řízení automobilu v bezproblémové situaci?
4. Kolik informací (např. předmětů) jsme si schopni zapamatovat v krátkodobé paměti?
5. O jakém fenoménu mluví Occamova břitva a jak jej můžeme využít při tvorbě uživatelského rozhraní?

Grafické uživatelské rozhraní

V následující kapitole se budeme věnovat obecným principům grafického uživatelského rozhraní a poté představíme základní prvky, které toto rozhraní tvoří.
Grafické uživatelské rozhraní poskytuje kvalitativní zlepšení uživatelské přívětivosti. Narozdíl od textového rozhraní umožňuje interakci formou grafických prvků jako jsou například tlačítka, formuláře a ikony.

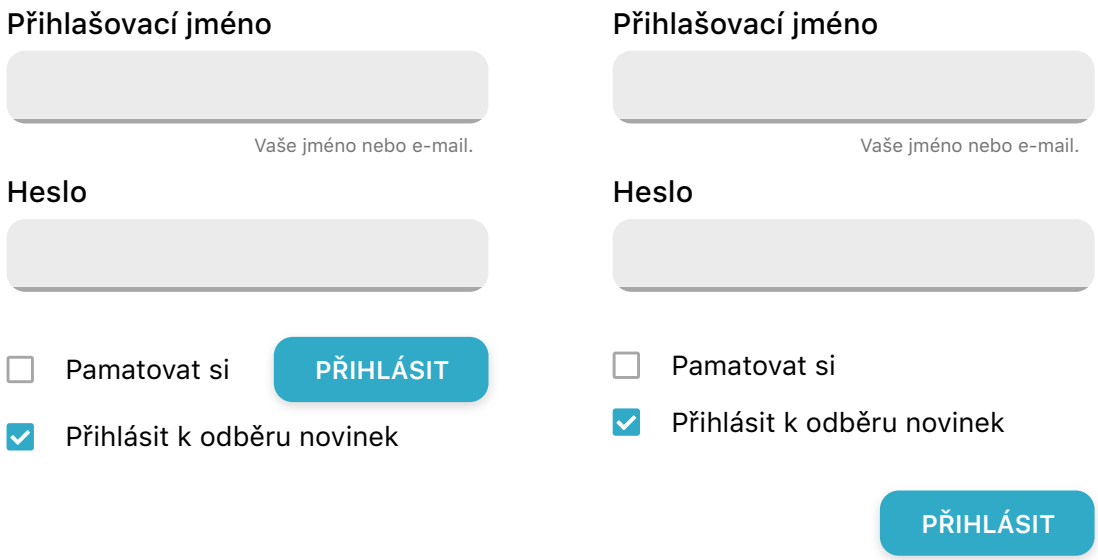
Průvodce studiem
V následující kapitole si ukážeme principy grafického uživatelského rozhraní. Ačkoli jsou již o něco více konkrétní než obecné principy z psychologie jsou stále dostatečně univerzální.

Kapitola čerpá z následujícího seznamu literatury, zájemcům o hlubší pochopení problematiky ji lze jenom doporučit:

- Jenifer Tidwell. Designing Interfaces, 2011 [13],
- Steve Krug. Don't Make Me Think, 2014 [8],
- Wilbert O. Galitz. The Essential Guide to User Interface Design, 2007 [1],
- Robin Williams. The Non-Designer's Design Book, 2014 [14],
- Everett N. McKay. UI is Communication, 2013 [9].

Grafický layout

Grafický layout neboli *rozložení* je umístění, velikost a vzdálenost prvků uživatelského rozhraní v rámci obrazovky. Layout komunikuje účel obrazovky, jakým způsobem ji číst a v jakém pořadí interagovat s jejími prvky.
Správné rozložení je většinou nejdůležitější a zároveň nejnáročnější součástí celého návrhu. Pro návrh grafického rozložení existuje sada pravidel, která jeho tvorbu usnadňují. Častou chybou je přílišný důraz na na vyplnění volného prostoru obrazovky. Layout není Tetris a umístění každého prvku musí mít svůj důvod (situace je demonstrována na obrázku 43).



Obrázek 43: Layout není Tetris, vlevo špatně, vpravo správně.

Dva způsoby prohlížení: čtení a skenování

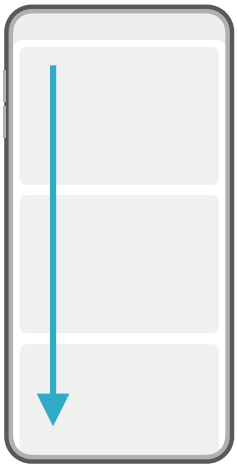
Steve Krug ve své knížce „Don't make me think" [8] upozorňuje, že uživatelé nečtou, uživatelé skenují. Nyní si vysvětlíme hlavní rozdíly.

Čtení Je podobné běžnému čtení textu, cílem je získat podrobné informace. V západní kultuře jsme zvyklí číst zleva doprava a během čtení přečteme většinu viditelného obsahu/textu. V případě, že nás nějaká pasáž nezaujme, přeskakujeme ji.

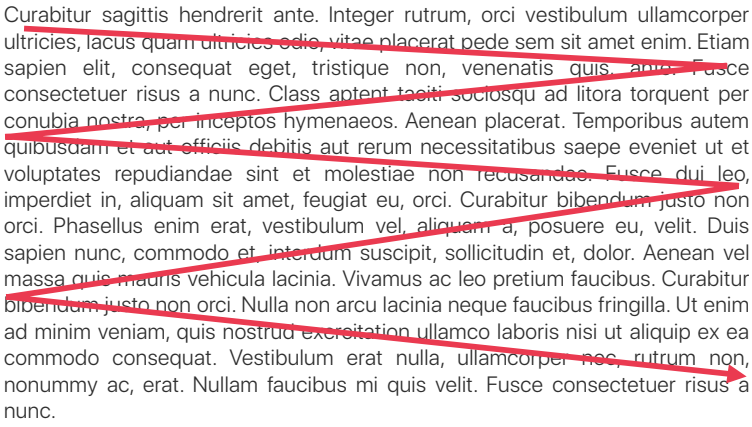
Skenování Pokud uživatel potřebuje najít konkrétní informaci, začíná takzvaným skenováním, čtení veškerého obsahu by bylo časově náročné. Skenování začíná v levém horním rohu, kopíruje pomyslnou křivku a končí v pravém dolním rohu. V případě menších zařízení (například mobilní telefon) je skenování jednoduší – uživatelé skenují obrazovku od horního okraje po spodní.

Uživatelé tyto dva vzory prokládají, situaci demonstrujeme na obrázku 47. Proces skenování probíhá dokud některý z prvků uživatele nezaujme, poté následuje fáze detailního čtení, která trvá do té doby než uživatel nenalezne potřebné informace nebo nezjistí, že zde nejsou. Tento proces se poté opakuje tak dlouho, než uživatel nenarazí na spodní okraj obrazovky. Následně se musí rozhodnout zda se posunout níže, nebo přejít na jinou obrazovku[12].

Obrázek 46: Styl skenování na menších obrazovkách.



[12] Z tohoto důvodu byl vynález nekonečného posouvání obrazovky na sociálních sítí geniální, uživatel nikdy nenarazí na konec obsahu.



Obrázek 44: Způsob čtení pro kultury, které čtou zleva doprava.

Uživatelé jsou netrpěliví

Uživatelé jsou netrpěliví a potřebují v co nejkratším čase najít hledanou informaci. Steve Krug používá připodobnění ke žralokům. Uživatelé jsou neustále v pohybu, nikdy se nezastaví. Skenování nám umožní lokalizovat tu část/funkcionalitu, která nás zajímá. Jako lidé jsme ve skenování velice dobří, tuto dovednou používáme při hledání čehokoli, ať už se jedná o cenu produktu nebo ztracené klíče.

Uživatelé rovněž nehledají optimální řešení. Mohlo by se zdát, že uživatelé skenují celou obrazovku aplikace a až poté se rozhodnout jakou akci provedou. Uživatelé skenují a jakmile dostanou pocit, že našli co hledají, nic neověřují a konají. To je způsobeno nedostatečnou penalizací ze strany aplikace. V případě zasílání peněz v internetovém bankovnictví jsou uživatelé mnohem rozvážnější (vědí, že penalizace za chybu je velká).

Úkol 9
Navštivte libovolnou webovou stránku a soustřeďte se na momenty, kdy přepínáte mezi skenováním a čtením. Stejné vyzkoušejte na mobilním telefonu nebo tabletu.

Návrh grafického rozložení

S ohledem na dostupné informace pojednávající o chování uživatelů můžeme přizpůsobit proces návrhu uživatelského rozhraní. Jedná se především o tyto body:

- Využití známých konvencí.
- Vytváření efektivních hierarchii.
- Rozdělení aplikace do jednoznačně definovaných oblastí.
- Prvky které slouží k interakci, by měli být očividně rozeznatelné.

Obrázek 45: Styl skenování na větších obrazovkách.

- Eliminovat rušivé prvky.
- Formátování textu pro snadné skenování.
- Vyváženost obsahu obrazovky.

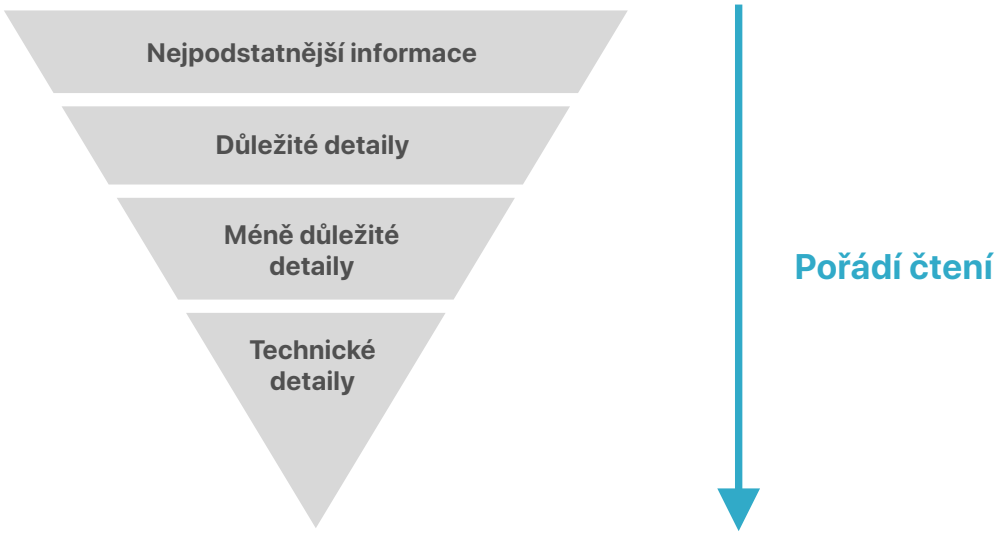
Využití známých konvencí Jeden z nejlepších způsobů jak učinit aplikaci pochopitelnou i ve spěchu je použití již známých konvencí. Konvencí myslíme to, jak prvky vypadají, jak se používají a k čemu slouží.

Vytváření efektivních hierarchií V rámci obrazovky je nutné vytvářet vizuální hierarchii. Nejdůležitější prvky by měly být největší, odsazené většími okraji a umístěny v horní části obrazovky. Prvky které spolu logicky souvisí by spolu měly souviset vizuálně.
Správná vizuální hierarchie vytváří strukturu obsahu, která znatelně ulehčuje skenování. *Invertovaná pyramida* symbolizuje správnou organizaci obsahu (obrázek 48). Při dodržení tohoto principu může uživatel čtení přerušit jakmile nalezne dostatečně konkrétní informace.

Úkol 10
Otevřete si detail produktu na libovolném e-shopu a ověřte zda je stránka správně navržena s ohledem na hierarchii informací.

Rozdělení aplikace do jednoznačně definovaných oblastí Rozdělení obrazovky na oblasti umožní uživatelům jejich jednoduché filtrování. Jakmile uživatelé naleznou užitečnou informaci, přestávají prozkoumávat ty ostatní. V oblastech, kde uživatelé očekávají reklamu, musíme být tedy opatrní. Je pravděpodobné, že tuto oblast budou ignorovat.

Obrázek 47: Čtení a skenování uživatel při práci s aplikací či webem kombinuje. Modrá barva značí skenování a červená čtení.



Obrázek 48: Invertovaná pyramida.

Jednoznačně rozeznatelné interaktivní prvky Prvky zprostředkovávající akci nebo navigaci musí být dobře viditelné. Uživatelé během skenování mimo jiné hledají interaktivní prvky. Všechny in-

teraktivní prvky by měli mít konzistentní podobu. Podobný princip lidé aplikují pokud hledají například kliku u dveří.

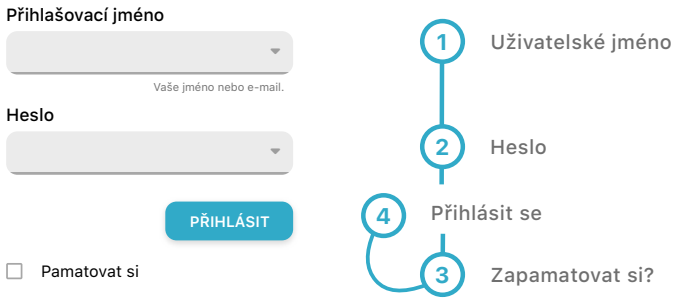
Eliminovat rušivé prvky Vizuální šum je velký nepřítel především webových stránek. Uživatelé mají rozdílný práh tolerance pro vizuálně komplexní weby. V kontextu urbanismu je tento jev nazýván *vizuální smog* – přehlcenost měst vizuálně nevhodnými billboardy/plakáty. S viditelnosti to tedy nesmíme přehánět.

Formátování textu pro snadné skenování V rámci uživatelského rozhraní stráví lidé nejvíce času prohledáváním a čtením textu. Je tedy vhodné text náležitě formátovat. Toho docílíme častým používáním nadpisů a rozlišování jejich úrovní. Text je vhodné rozdělovat do kratších odstavců obsahující ucelené informace. Skenování rovněž usnadňují odrážkové/číslované seznamy.

Pořadí prvků na stránce Pořadí prvků na obrazovce by mělo následovat přirozenou konverzaci. Nejvíce je to viditelné na příkladu formulářů:

- **Aplikace:** Jaké je Vaše uživatelské jméno?
- **Uživatel:** František Palacký
- **Aplikace:** Jaké je Vaše heslo?
- **Uživatel:** NojhwGuzjy
- **Aplikace:** Přejete si zůstat přihlášen?
- **Uživatel:** Ano

Obrázek 49: Správné pořadí prvků respektující přirozenou komunikaci.



Obrázek 50: Špatné pořadí prvků, nerespektuje přirozenou komunikaci.

Vyváženost Pomyslné těžiště obrazovky by mělo být v její polovině, v takovém případě jsou prvky uživatelské rozhraní rozmístěny rovnoměrně a obrazovka působí vyváženým dojmem. Příklad je viditelný na obrázku 51. Nesmíme však zapomínat, že cílem není vyplnit veškerý prázdný prostor – nezapomínejme na všechna ostatní pravidla.

Obrázek 51: Vyváženost layoutu.

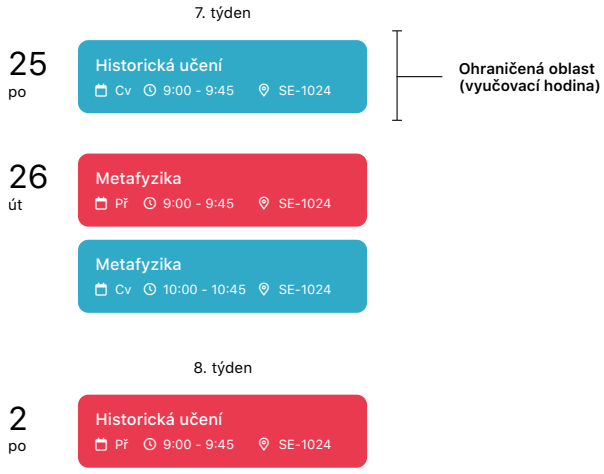
Principy tvarové psychologie

K dosažení srozumitelného rozvržení nám rovněž můžou pomoci následující tři principy vycházející z tvarové psychologie.

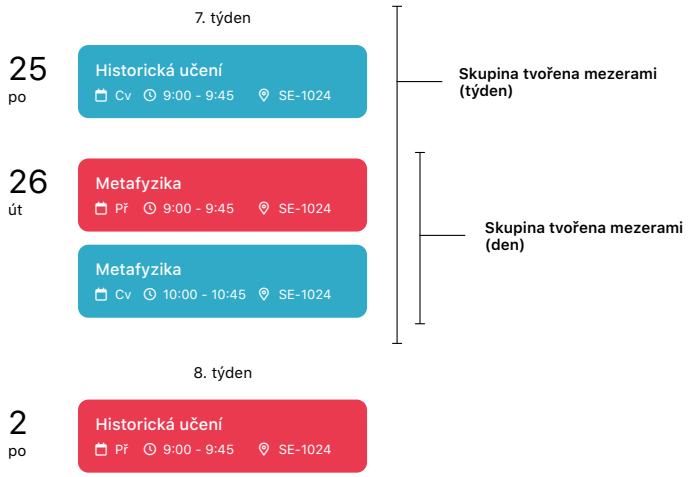
Zákon seskupení Prvky umístěné v ohraničené oblasti působí jako souvislá skupina. Tento jev je viditelný na obrázku 52. Uživatel dostává jasné vodítko, že se jedná o skupinu souvisejícího obsahu.

Zákon blízkosti Prvky umístěné blízko sebe (vhodně oddělené mezerou od okolního obsahu) tvoří skupinu. Příklad si ukážeme v rámci stejné obrazovky na obrázku 53.

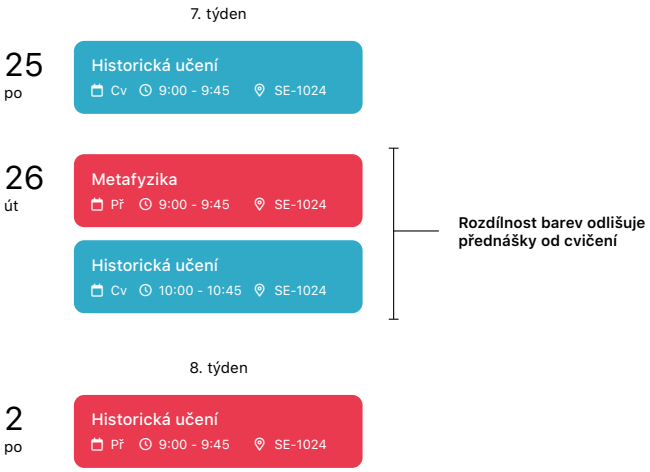
Zákon podobnosti Prvky vizuálně podobné tvoří skupinu bez ohledu na jejich umístění. Naopak vizuální rozdílnost může uživatele upozornit na odlišnost prvků v kontextu jejich okolí. Tento princip můžeme využít pro spojení prvků které nelze spojit pomoci zákonu blízkosti. Situace je viditelná na obrázku 54.



Obrázek 52: Demonstrace zákonu seskupení.



Obrázek 53: Demonstrace zákonu blízkosti.



Obrázek 54: Demonstrace zákonu podobnosti.

Mobile-first design

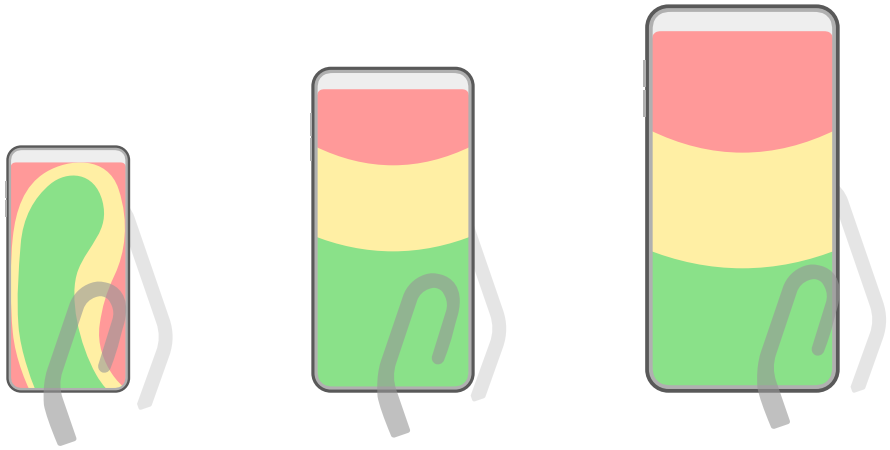
Jedná se o přístup k návrhu uživatelského rozhraní, při kterém nejdříve navrhujeme rozložení pro menší obrazovky mobilních zařízení. Tento přístup má hned několik výhod. Podíl uživatelů kteří používají k práci s aplikací výhradně mobilní telefon stále roste. Dávno již tedy neplatí, že telefony tvoří zanedbatelná procenta uživatelů.
Pokud se při návrhu zaměříme nejdříve na mobilní zařízení, budeme nuceni upřednostňovat jednoduchost před komplexností. Rovněž je vždy jednoduší upravit mobilní rozhraní na to plnohodnotné (prvky přidávat) než je tomu naopak (prvky odebrat).

Mobile-first design tedy zlepšuje:

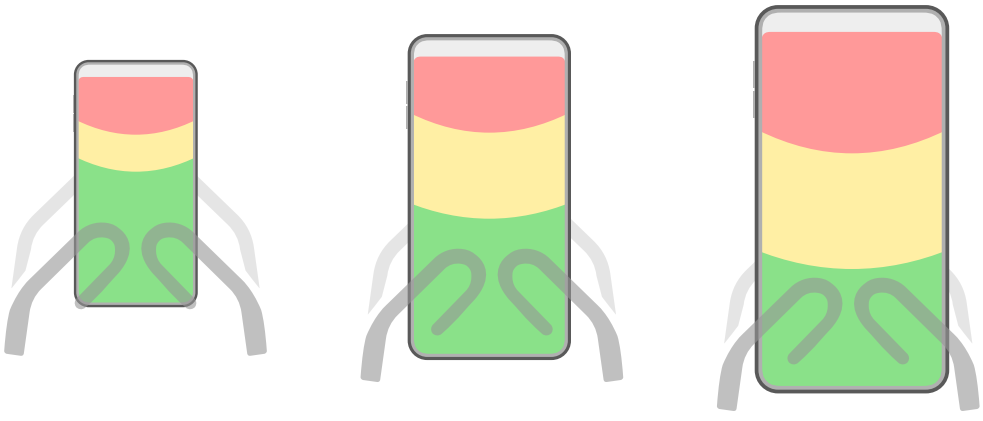
- Srozumitelnost a přehlednost.
- Zavádí snadnější lineární způsob konzumace obsahu.
- Zvětšuje velikost ovládacích prvků.

Způsob držení mobilních zařízení

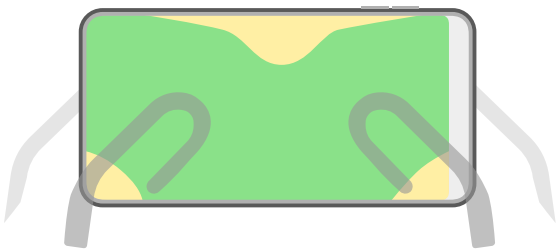
To jakým způsobem uživatelé drží svá mobilní zařízení má velký vliv na styl a možnost jeho ovládání. Běžný uživatel průběžně střídá různé úchopy a orientace mobilního zařízení.
Většina uživatelů drží telefon v jedné ruce a dotykovou obrazovku ovládá palcem (obrázek 55), pouze 28 % uživatelů drží telefon v jedné ruce a dotykovou obrazovku ovládá ukazováčkem druhé ruky. Zelená oblast je dosažitelná nejlépe, žlutá hůře a červená nejhůře.
Při držení telefonu oběma rukama je situace o něco lepší (obrázek 56). Toto držení uživatelé používají nejčastěji při psaní.
Orientace na šířku (obrázek 57) je nejčastěji používaná při hraní her nebo sledování multimediálního obsahu. V této orientaci je dosažitelná skoro celá obrazovka.
U větších zařízení (jako je například tablet) je nejčastější držení oběma rukama (obrázek 58).



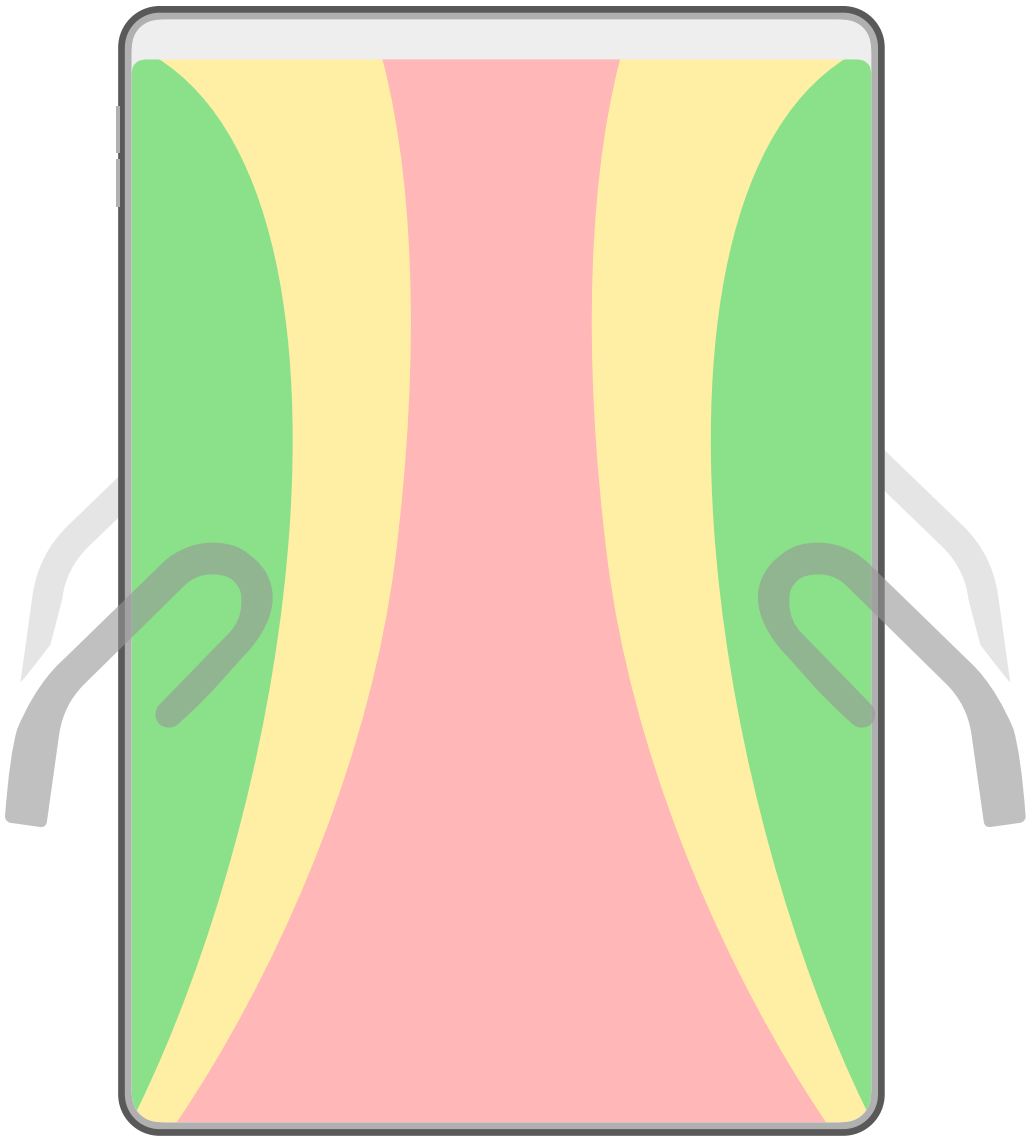
Obrázek 55: Demonstrace ovladatelnosti mobilního telefonu v orientaci na výšku.



Obrázek 56: Demonstrace ovladatelnosti mobilního telefonu drženého oběma rukama (orientace na výšku).

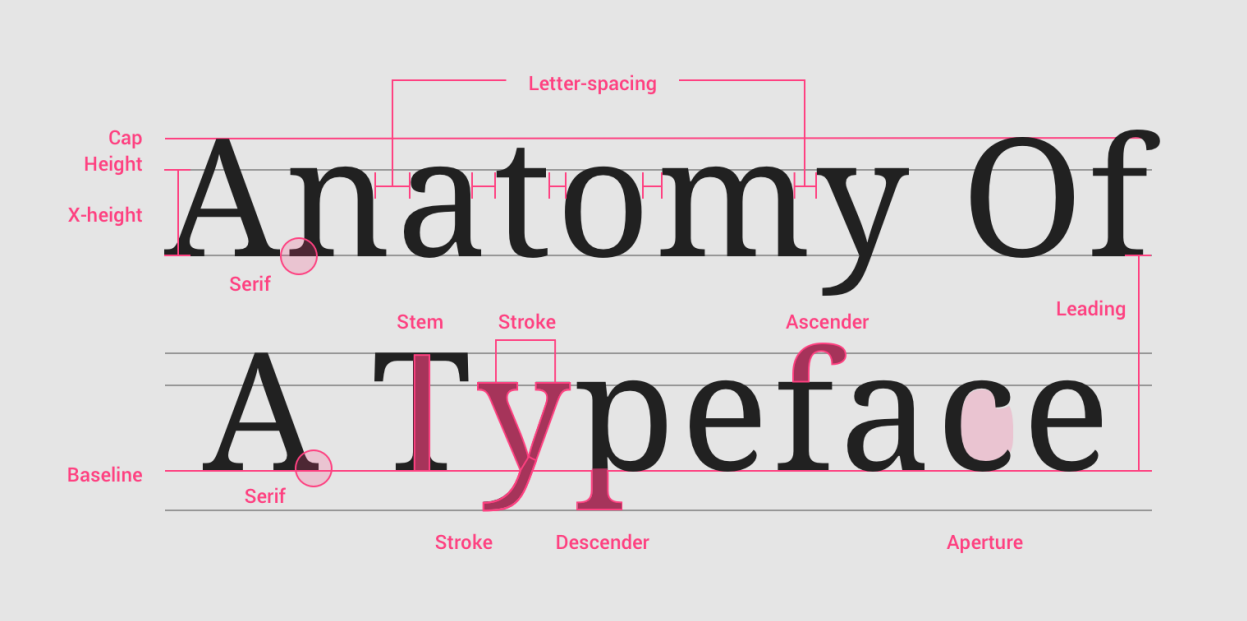


Obrázek 57: Demonstrace ovladatelnosti mobilního telefonu drženého oběma rukama (orientace na šířku).



Obrázek 58: Demonstrace ovladatelnosti tabletu drženého oběma rukama v orientaci na výšku.

Typografie



Obrázek 59: material.io – Understanding typography.

„I decided to take a calligraphy class to learn how to "learn calligraphy". I learned about serif and sans-serif typefaces, about varying the space between different letter combinations, about what makes great typography great. It was beautiful. Historical. Artistically subtle in a way that science can't capture. And I found it fascinating. None of this had any hope of any practical application in my life. But 10 years later, when we were designing the first Macintosh computer, it all came back to me. And we designed it all into the Mac. It was the first computer with beautiful typography. If I had never dropped in on that single course in college, the Mac would never have multiple typefaces or proportionally spaced fonts. And since Windows just copied the Mac, it's likely that no personal computer would have them."

Steve Jobs,

V následující sekci se budeme okrajově věnoval typografii [2]. Typografie je umění stylu, vzhledu a struktury písma se záměrem poskytnout čtenáři estetický a dobře čitelný text. Můžeme říct, že se jedná o transformaci jazyka do dekorativního vizuálního elementu. Historie typografie je poměrně bohatá, začíná někde okolo 11–12. století.

Průvodce studiem
S typografii není většinou potřeba významně experimentovat, obvykle stačí vybrat jedeno nebo dvě vhodná písma.

Proč je typografie důležitá? Většina vizuální krásy designu pramení právě z písma. Špatně provedená typografie má znatelný dopad na kvalitu designu uživatelského rozhraní[13].
Klíčové pojmy z typografie budeme uvádět v anglickém jazyce, jelikož je to nejčastější forma jejich výskytu v příručkách/softwaru.

[13] Jak esteticky, tak funkčně – například v čitelnosti textu

Font vs. Typeface

Font a *typeface* jsou často zaměňované pojmy. Typeface si můžeme představit jako „rodinu“ a jednotlivé fonty jsou její členové. Typeface je tedy styl designu písma, který obsahuje kompletní sadu znaků ve všech velikostech a tloušťkách. Font je potom grafická reprezentace textového znaku konkrétní typeface, velikosti a tloušťky. Ve většině případu[14] jsou tyto pojmy již volně zaměňovány.

[14] Převážně s nástupem digitální typografie

Mean vs. Base line

Znaky textu jsou nejčastěji umístěny na pomyslné rovné lince – řádku. Tuto linku nazýváme *base line* – značí spodní okraj těla znaku. V kontrastu *mean line* značí horní okraj těla znaku. Celá situace je ilustrována na obrázku 60.

Obrázek 60: Mean line vs. Baseline.

Size, weight, height

Vlastnosti umožňující designérům rozlišit důležitost různých informací.

Size *Size* neboli velikost písma, uváděno v různých jednotkách „px“, „pt“, „em“ a další.

Weight *Weight* neboli tloušťka písma, určena číselnou hodnotou 100, 200, 300, atd., případně hodnotami „light“, „medium“, „bold“ a další.

Průvodce studiem
Velikost a tloušťka patří spolu s řádkováním mezi nejdůležitější vlastnosti vysázeného textu, na tyto vlastnosti bychom se měli vždy zaměřit.

Height *Height* nebo také *x-height*, česky pak výška znaku. Odvíjí se od znaku písmena „x“, viditelné na obrázku 61.

Obrázek 61: X-height.

Ascender vs. Descender

Ascender je část znaku sahající nad mean line (například u znaku „b“ nebo „h“). *Descender* je část znaku sahající pod baseline (například u znaku „g“ nebo „j“). Rozdíl je viditelný na obrázku 62.

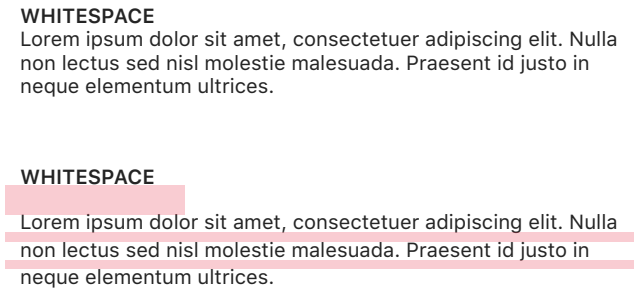
Obrázek 62: Ascender vs. Descender.

Whitespace

Whitespace nebo také *negative space* je volná plocha mezi grafickými prvky. Ačkoli je whitespace neviditelný, ovlivňuje zásadním způsobem čitelnost textu.

Alignment

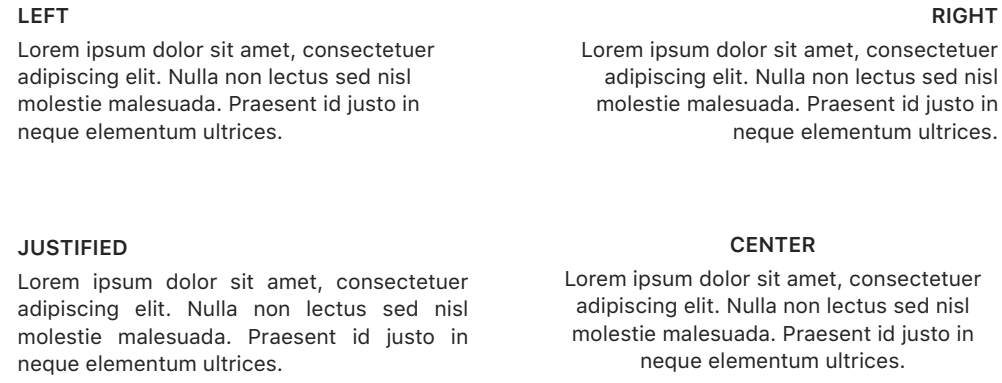
Alignment neboli *zarovnání* určuje k jakému okraji (případně středu) je text zarovnaný. Umožňuje spojit jinak nespojitý text do bloku. Existují tři druhy zarovnání: „left“ (vlevo), „center“ (na střed), „right“ (vpravo), „justified“ (do bloku). Jednotlivá zarovnání jsou demonstrována na obrázku 64.



Obrázek 63: Whitespace (vyznačený červeně).

Nejvíce problematické je zarovnání do bloku (justified). Z důvodu vynuceného zarovnání a mohou vznikat takzvané *řeky* (nejlépe pochopitelné z obrázku 65). Tento nedostatek lze vyřešit zalamováním slov, poté je možné zarovnání justified používat bez obav[15].
Obecně ale platí, že ve většině případů bychom si měli vystačit se zarovnáním vlevo.

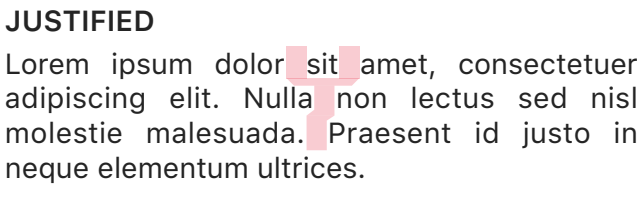
[15] Například text, který právě čtete využívá zarovnání justified se zalamováním slov.



Obrázek 64: Demonstrace typů zarovnání.

Tracking vs. Kerning

Tracking je proces upravování globální velikosti mezery mezi jednotlivými znaky. Správná volba velikosti mezery učiní text vzdušným a dobře čitelným.



Obrázek 65: Demonstrace „řeky“ v případě zarovnání justified.

Obrázek 66: Demonstrace tracking a kerningu.

Kerning je proces podobný trackingu, jedná se však o upravování mezery mezi dvěma konkrétními znaky. Jedná se tedy o individuální zásah, který přispěje k čitelnějšímu nebo esteticky působivějšímu textu.

Leading

Leading neboli *řádkování* určuje velikost mezery mezi dvěma baseline. Správně nastavené řádkování napomáhá snadnému čtení textu. Standardní velikost řádkování je 120 % velikosti písma.

Serif vs. Sans-serif

Hlavní rozdíl je patrný na první pohled, *serif* neboli *patková* písma obsahují takzvané *patky* (viditelné na obrázku 68). *Sans-serif* písma patky neobsahují.
Dříve platilo, že patková písma nacházela místo především v tiskovinách, neprokázalo se však, že by byla lépe čitelná než písma bezpatková. Důvodem bylo, že starší typy obrazovek nedisponovaly dostatečně vysokým rozlišením pro vykreslení často malých patek písma. V dnešní době obrazovek s vysokým rozlišením se tedy jedná o volbu založenou na preferenci designéra [6].

Obrázek 67: Demonstrace leading.

Obrázek 68: Serif písmo se zvýrazněnými patkami.

Navigace

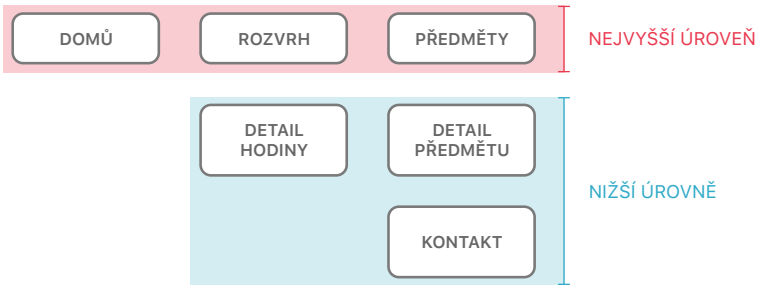
Navigace je pohyb mezi jednotlivými obrazovkami aplikace/webu za účelem dosáhnout stanoveného cíle. V následující sekci nastíníme základní principy navigace popsané v příručce Material Designu od firmy Google [3].

Průvodce studiem
Navigace patří mezi nejdůležitější části celé aplikace. Před návrhem jednotlivých obrazovek vždy začínejte s návrhem navigace, jaké obrazovky bude aplikace obsahovat a jak se mezi nimi bude uživatel pohybovat.

Vrstvy aplikace

Každá aplikace obsahuje několik vrstev, které jsou tvořeny jednotlivými obrazovkami. Navigace probíhá mezi jednotlivými obrazovkami.[16]. Příklad vrstev aplikace je viditelný na obrázku 69

[16] Ať už v jedné vrstvě nebo napříč vrstvami



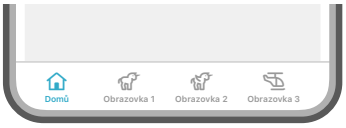
Obrázek 69: Jednotlivé vrstvy aplikace.

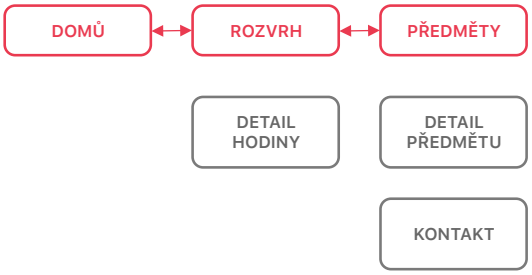
Typy navigace

Typy navigace určují způsob přechodu mezi obrazovkami. Rozlišujeme zda uživatel sestupuje z jedné vrstvy do druhé, nebo zda se pohybuje v rámci jedné vrstvy.

Boční navigace Navigace mezi hlavními obrazovkami aplikace. Často rovněž označována jako *hlavní navigace*. K pohybu dochází pouze na nejvyšší úrovni aplikace (demonstrováno na obrázku 71).

Obrázek 70: Hlavní navigace v dolní části obrazovky.

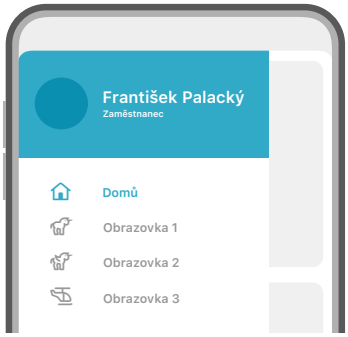




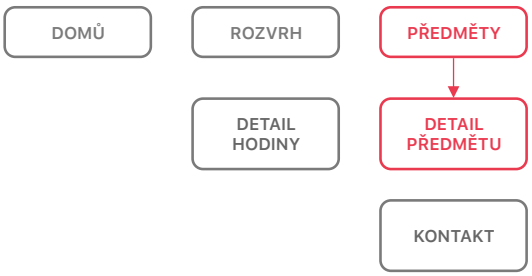
Obrázek 71: Boční navigace v rámci hlavních obrazovek.

Hlavní navigaci obvykle realizuje navigačním menu, to může mít několik podob, příklady nalezneme na obrázcích 70 a 72. Navigační lišta na spodní straně mobilního zařízení je dnes upřednostňovanou možností.

Obrázek 72: Hlavní navigace vysouvající se z boku obrazovky.

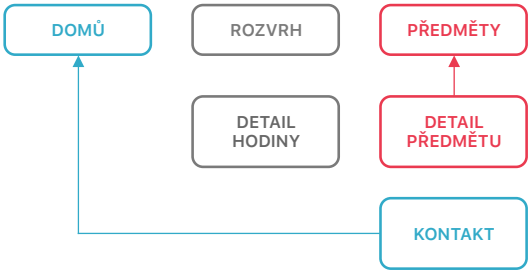


Dopředná navigace Pohyb mezi obrazovkami po sobě jdoucích vrstvách (obrázek 73), jednotlivé vrstvy je možné přeskakovat. Dopřednou navigace realizujeme tlačítky, obrázky a dalšími prvky.



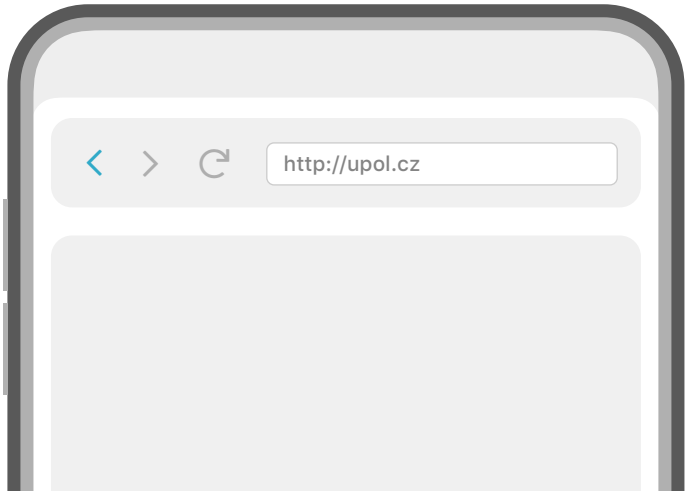
Obrázek 73: Dopředná navigace mezi obrazovkami.

Zpětná navigace Zpětný pohyb v aplikaci u kterého rozlišujeme dva druhy: *chronologická* a *hierarchická*. Chronologická zpětná navigace zachovává pořadí obrazovek kterými uživatel přišel. Hierarchická zpětná navigace postupuje zpět po jednotlivých vrstvách, které nepřeskakuje.

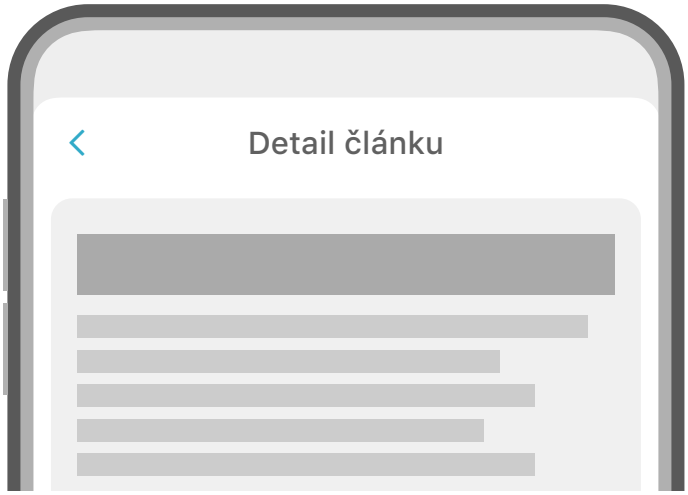


Obrázek 74: Zpětná navigace mezi obrazovkami.

Implementace zpětné navigace je platformově závislá, v drtivé většině je však realizována některou formou „šipky zpět“. Příklad chronologické navigace je možné vidět na obrázku 75. Obrázek 76 demonstruje navigaci hierarchickou.



Obrázek 75: Chronologická zpětná navigace, příklad z webového prohlížeče.



Obrázek 76: Hierarchická zpětná navigace, příklad zobrazení detailu článku a následné vrácení zpět.

Shrnutí

Grafický layout, typografie a správně navržená navigace patří mezi základy každého dobrého uživatelského rozhraní. Dozvěděli jsme se, jakým způsobem uživatelé prohlíží obsah obrazovky, získali přehled o základních pojmech z typografie a víme jaké typy navigace můžeme v rámci aplikace používat.

Kontrolní otázky
Odpovězte na následující otázky:

1. Jakou akci provádí uživatel zpravidla jako první, skenování nebo čtení?
2. Co můžeme udělat pro snadnější skenování uživatelského rozhraní?
3. Co je mobile-first design?
4. Jaké typy navigace si pamatujete? Není podstatné znát jejich název, stačí je umět rozlišit a chápat rozdíly.

Prvky grafického uživatelského rozhraní

V předchozí kapitole jsme si představili základní principy grafického uživatelského rozhraní. V této kapitole podrobně představíme jednotlivé prvky (stavební kameny).
Názvy prvků si uvedeme primárně v anglickém jazyce. Jednotlivé prvky se pokusíme popsat co nejobecněji. Je tedy možné, že se v praxi můžeme setkat s drobnými odlišnostmi.

Průvodce studiem
Konkrétní prvky uživatelského rozhraní patří v procesu tvorby UI mezi nejčastěji se měnící elementy. Doporučuji vždy prostudovat pravidla pro danou platformu a sledovat nejnovější trendy.

Příkazy a akce

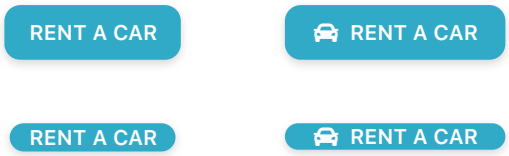
V následující sekci se budeme věnovat prvkům reprezentující příkazy a akce. Některé zde uvedené prvky mohou sloužit k několika účelům, rozdělení je tedy pouze orientační.

Button

Button neboli *tlačítko* je jeden z nejzákladnějších prvků uživatelského rozhraní. Jedná se o jasný prvek interakce, kterému většina uživatelů rozumí. To je způsobeno tím, že se tlačítka vyskytují všude kolem nás.
Tvarově je nejlepší následovat příklady z reálného světa. Typické tlačítko je hranaté nebo kulaté. Zaoblené rohy přitahují uživatelovu pozornost [12] a působí přívětivěji.
Největším problémem tlačítek je jejich viditelnost (obrázek 80). Jak již bylo zmíněno, tento problém začal být aktuálnější s příchodem flat designu. Tlačítka bez výplně, případně s pouhým rámečkem (takzvané *ghost button*) mohou negativně ovlivňovat viditelnost (obrázek 78).
Mimo text může tlačítko obsahovat ostatní signifiery (například ikonu). V jakékoli variantě tlačítka je důležité dodržovat rozumné vnitřní okraje (takzvaný *padding*, obrázek 79).
Tlačítka se mohou vyskytovat v několika stavech. Běžně se uvádí těchto pět stavů: default, hover, focus, pressed, disabled (obrázek 81).

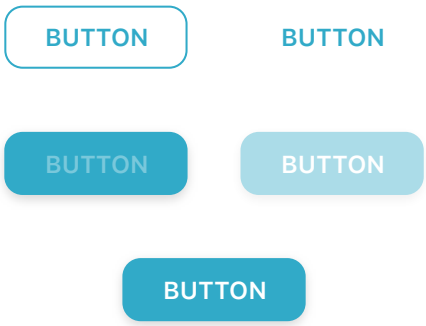
Obrázek 77: Tlačítko jako jasný prvek interakce.

Obrázek 78: Ghost button vs. typické tlačítko z Flat designu.

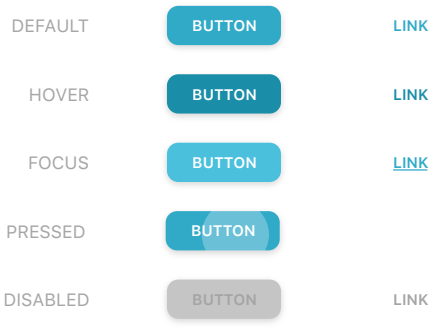


Obrázek 79: Příklad tlačítek bez ikony/s ikonou a špatným/správným paddingem.

- **default** – Výchozí stav tlačítka.
- **hover** – Zvýraznění tlačítka po najetí kurzoru.
- **focus** – Tlačítko po označení (například klávesnicí – důležité pro přístupnost).
- **pressed** – Stav tlačítka při zmáčknutí (důležité pro dotykové obrazovky).
- **disabled** – Stav vypnutého tlačítka, které nelze aktivovat.



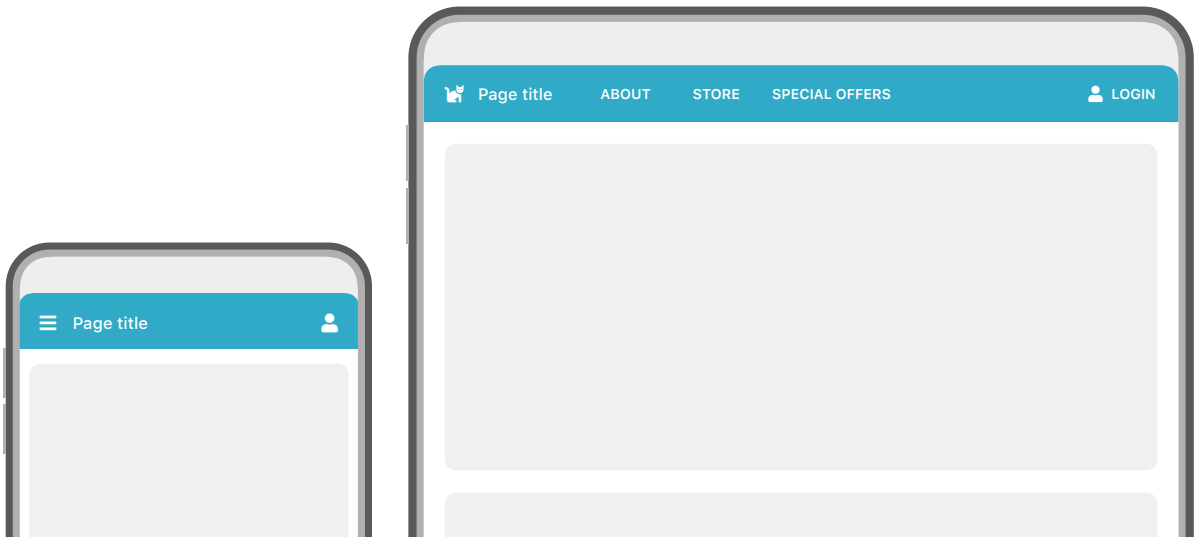
Obrázek 80: Tlačítka s různou viditelností, které jsou podle vás nejlépe a nejhůře viditelná/rozeznatelná?



Obrázek 81: Jednotlivé stavy tlačítka.

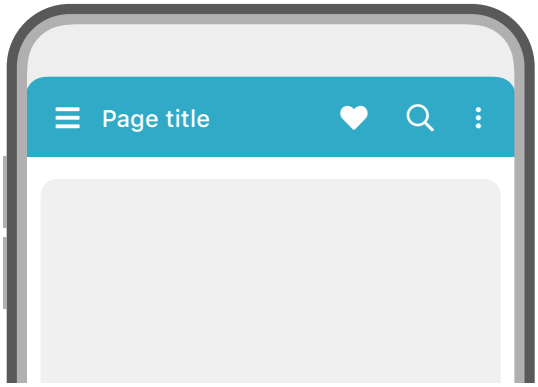
Navigation bar

Navigation bar neboli *navigační lišta* slouží k hlavní navigaci v rámci aplikace.
Lišta bývá zpravidla umístěna na horním okraji a její pozice může být fixovaná – při scrollování zůstává připnuta na horní okraj obrazovky. Navigační lišta může měnit svoji podobu na základě kontextu, obecně by však měla být konzistentní napříč aplikací.

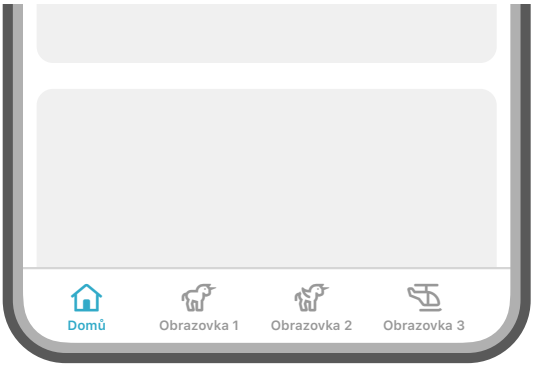


Obrázek 82: Navigation bar na zařízeních různé velikosti.

V některých případech může navigační lišta obsahovat akce spojené s aktuální obrazovkou. Jedná se tak o spojení navigační lišty a lišty nástrojů (o ní si povíme později). Toto spojení je časté u mobilních telefonů a zařízení s menší obrazovkou.

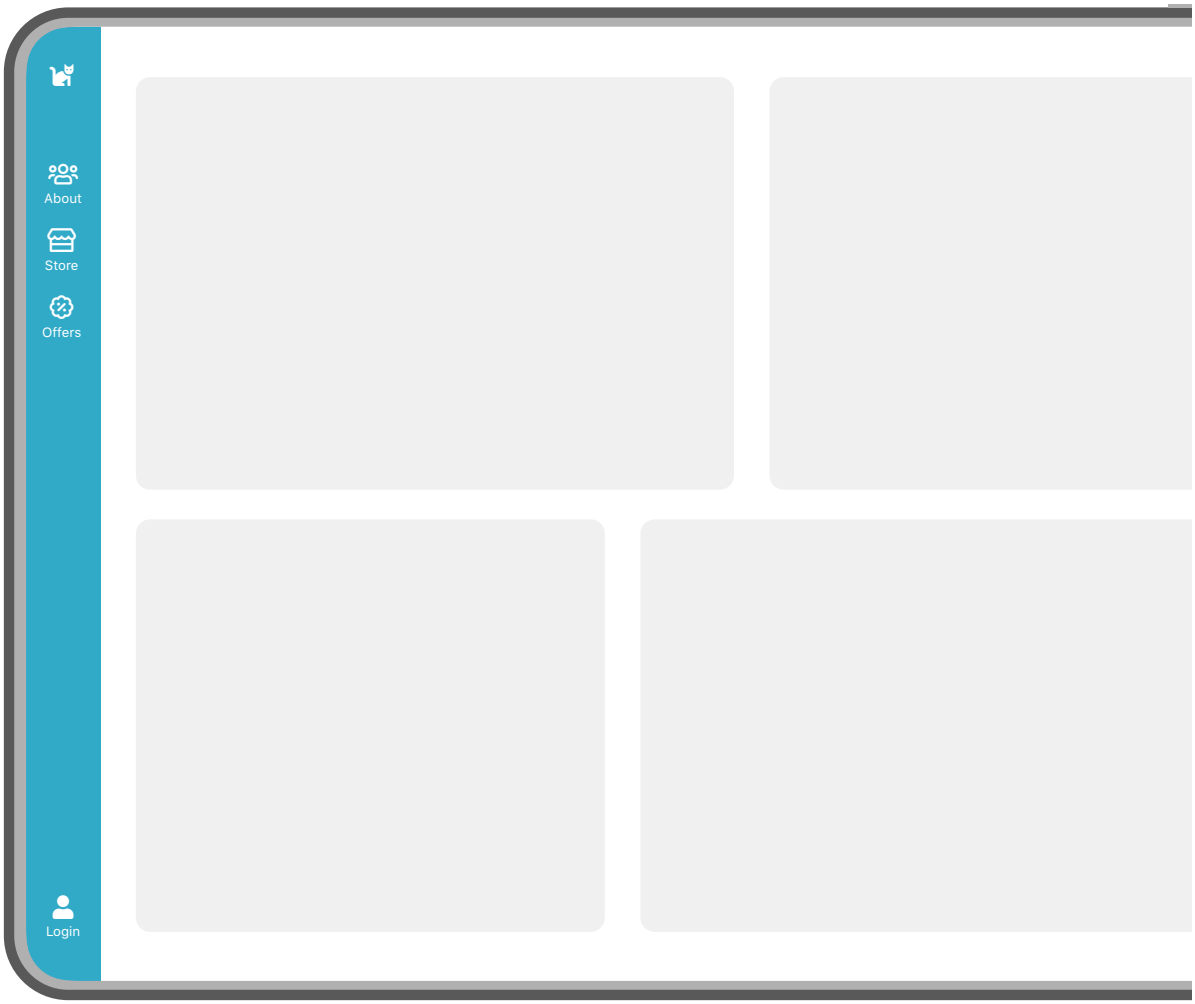


Obrázek 83: Spojení navigační lišty a lišty nástrojů.



Obrázek 84: Navigační lišta umístěná v dolní části obrazovky mobilního telefonu.

U širokých obrazovek lze využít vertikální varianty. V takovém případě je však nutné využít ikon (delší text se na šířku lišty nevleze). Tato varianta je viditelná na obrázku 85.

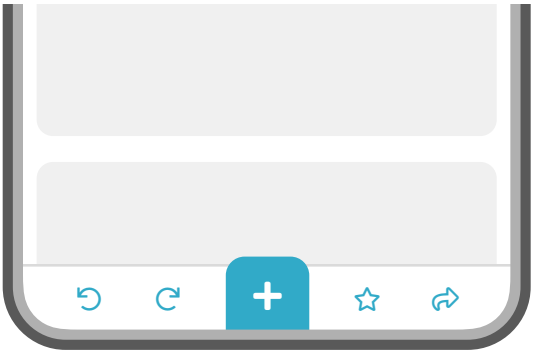


Obrázek 85: Vertikální varianta navigační lišty.

Tool bar

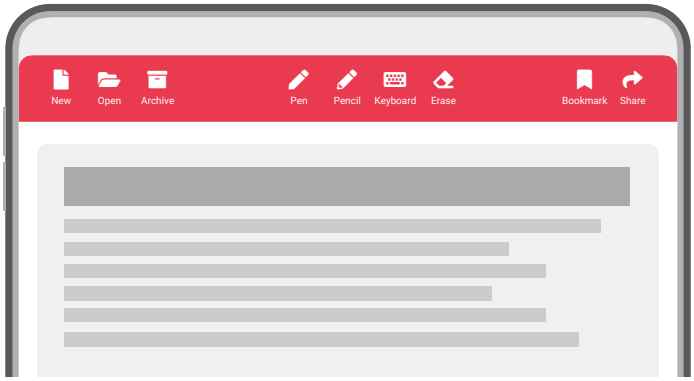
Tool bar neboli *lišta nástrojů* poskytuje akce dostupné pro aktuální kontext aplikace. Akce symbolizujeme kombinací textu a ikony. Podobně jako navigační lišta je umísťován do horní části obrazovky. Není to však striktním pravidlem. Především na telefonech je často možné lištu nástrojů vidět na spodním okraji obrazovky (obrázek 86)[17]. Na webu se pak lišta nástrojů může vyskytovat kdekoli na stránce.

[17] Dolní pozice lišty nástrojů je postupně vytlačena navigační lištou umístěnou ve spodní části obrazovky mobilního telefonu



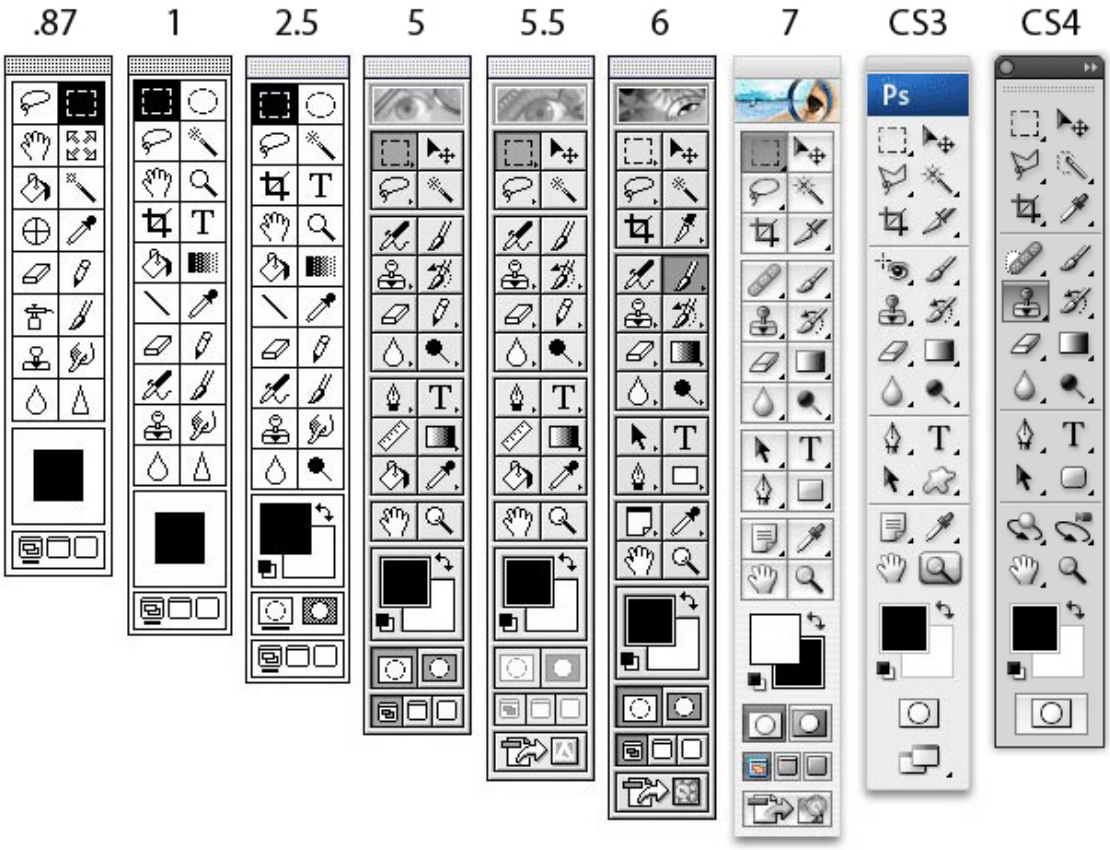
Obrázek 86: Lišta nástrojů na mobilním telefonu.

V rámci lišty nástrojů je vhodné prvky rozdělovat do skupin po 3 až 7 prvcích dle Millerova pravidla uvedeného dříve (obrázek 87).



Obrázek 87: Horizontální lišta nástrojů respektující Millerovo pravidlo.

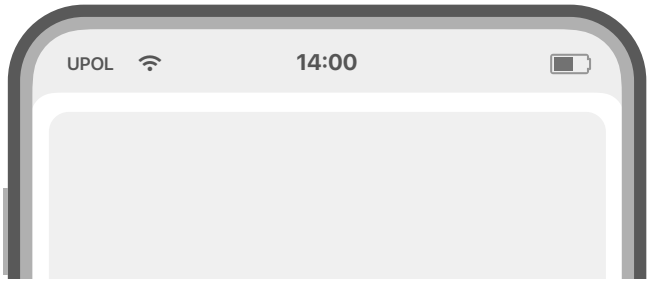
Časté je rovněž použití vertikální lišty nástrojů (příklad z programu Photoshop viditelný na obrázku 88).



Obrázek 88: Vertikální lišta nástrojů z programu Photoshop.

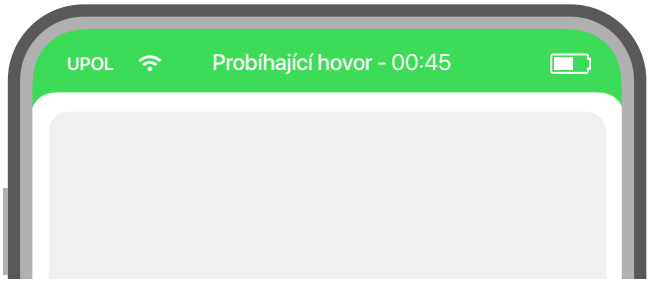
Status bar

Status bar neboli *stavová lišta* obsahuje aktuální informace o aplikaci/zařízení nebo probíhající operaci. Může se nacházet jak v horní (mobilní zařízení) tak dolní (web a desktop aplikace) části obrazovky. Rovněž může obsahovat prvky sloužící k interakci.



Obrázek 89: Stavová lišta na mobilních zařízeních.

Stavová lišta může měnit barvu na základě probíhající akce (obrázek 90), případně nemusí být zobrazena trvale (obrázek 91).

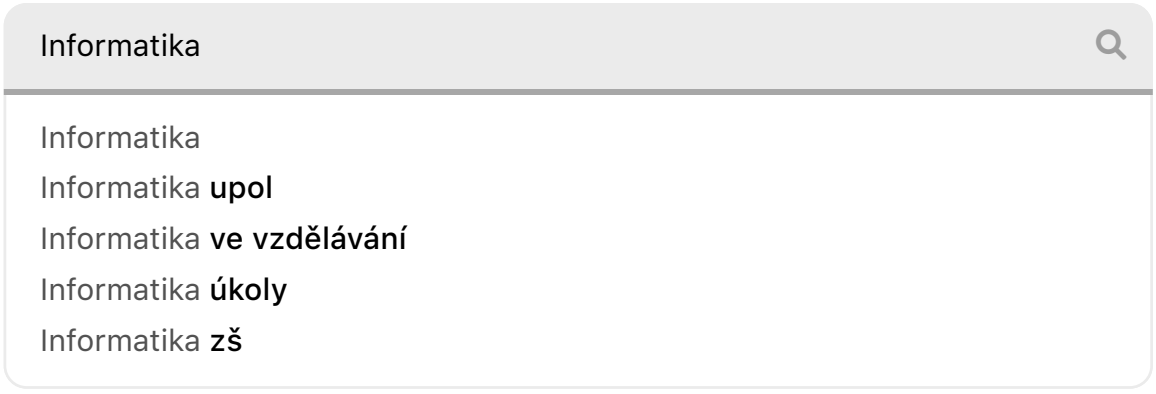


Obrázek 90: Stavová lišta během probíhajícího hovoru.

Obrázek 91: Stavová lišta, která je zobrazena pouze když je potřeba (náhled cíle odkazu).

Search bar

Search bar neboli *vyhledávací lišta* slouží ke zprostředkování vyhledávání v rámci aplikace. V principu se jedná o modifikovaný textový vstup. Vyhledávací lištu však uvádíme jako samostatný prvek z důvodu jejího rozšíření a dodatečné funkcionality.
Jak už bylo řečeno, tvarově se jedná o klasický textový vstup, který standardně obsahuje ikonu a text. V některých případech je vyhledávací lišta spojena s potvrzovacím tlačítkem, které zahajuje vyhledávání. Často tuto funkci doplňuje klávesa „Enter“.

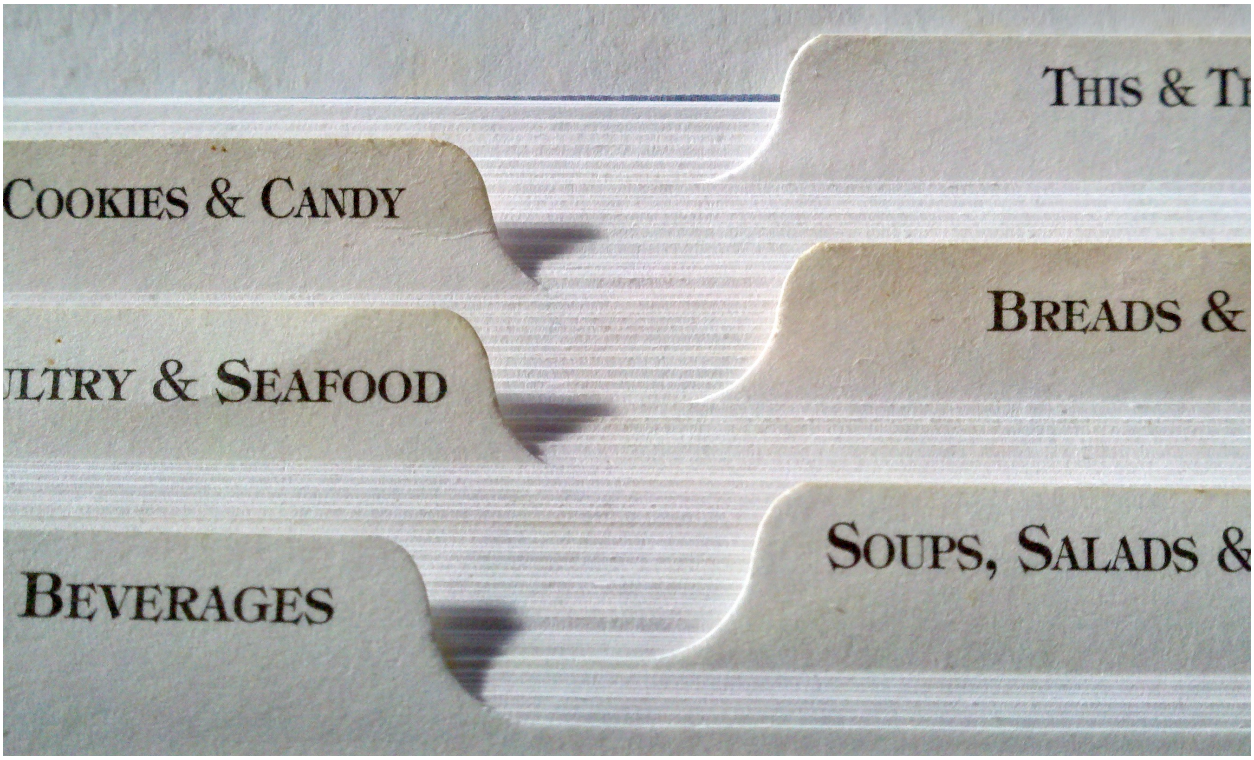


Obrázek 92: Vyhledávací lišta s našeptáváním.

Důležitou přidanou funkcionalitou je našeptávání hledaných frází. Našeptávání výrazně zlepšuje použitelnost a snižuje zátěž na paměť uživatele.
Lišta vyhledávání se může nacházet kdekoli v aplikaci, nejčastěji však najde své místo v rámci navigační lišty nebo lišty nástrojů.

Tab bar

Tab bar neboli *lišta záložek* slouží k přepínání obsahu ve vymezené oblasti. Touto oblastí můžou být celé obrazovky nebo jejich části. Často rovněž využívaná pro potřeby navigace (obrázek 84). Tvarově inspirováno složkami v kartotéce (obrázek 93).
Uživatel je však vždy odkázán pouze na popis záložek, obsah vidí až po jejich zobrazení. Tato nevýhoda je kompenzovaná velkou úsporou místa.
Pro bezproblémovou práci s tab barem je nutné dostatečně viditelně indikovat právě zvolenou záložku.



Obrázek 93: Předloha pro záložky v uživatelském rozhraní.

Progress bar/indicator

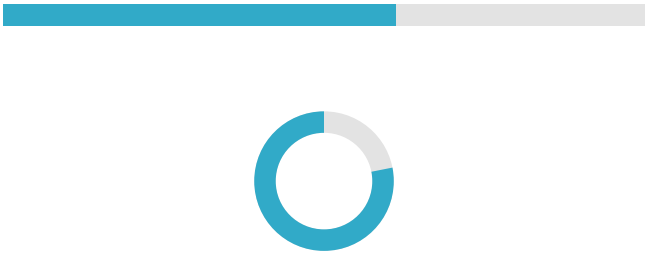
Progress bar/indicator neboli *ukazatel průběhu* slouží k indikaci právě probíhající akce. Tyto dvě varianty je nutné důsledně rozlišovat. Tvarově je možné použít jak lineární tak kruhovou variantu.

Progress bar Slouží k indikace průběhu akce u které známe alespoň přibližnou dobu trvání. Odhad doby trvání by měl být vždy co nejlepší. V případě, že nejsme schopni tuto přesnost zaručit je lepší použít progress indicator.

Progress indicator Slouží k indikaci průběhu akce u které neznáme dobu trvání. Ve většina případů má podobu neustále se opakující animace (například přesýpací hodiny, točící se kolečko a další). Pokud doba čekání přesáhne obvyklou dobu, je vhodné uživatele informovat, že aplikace stále pracuje (nezasekla se).

Drop-down list

Drop-down list neboli *rozbalovací seznam* má mnoho využití. Obecně slouží k výběru z několika hodnot seznamu. Obvykle se používá ve formulářích nebo v navigaci.



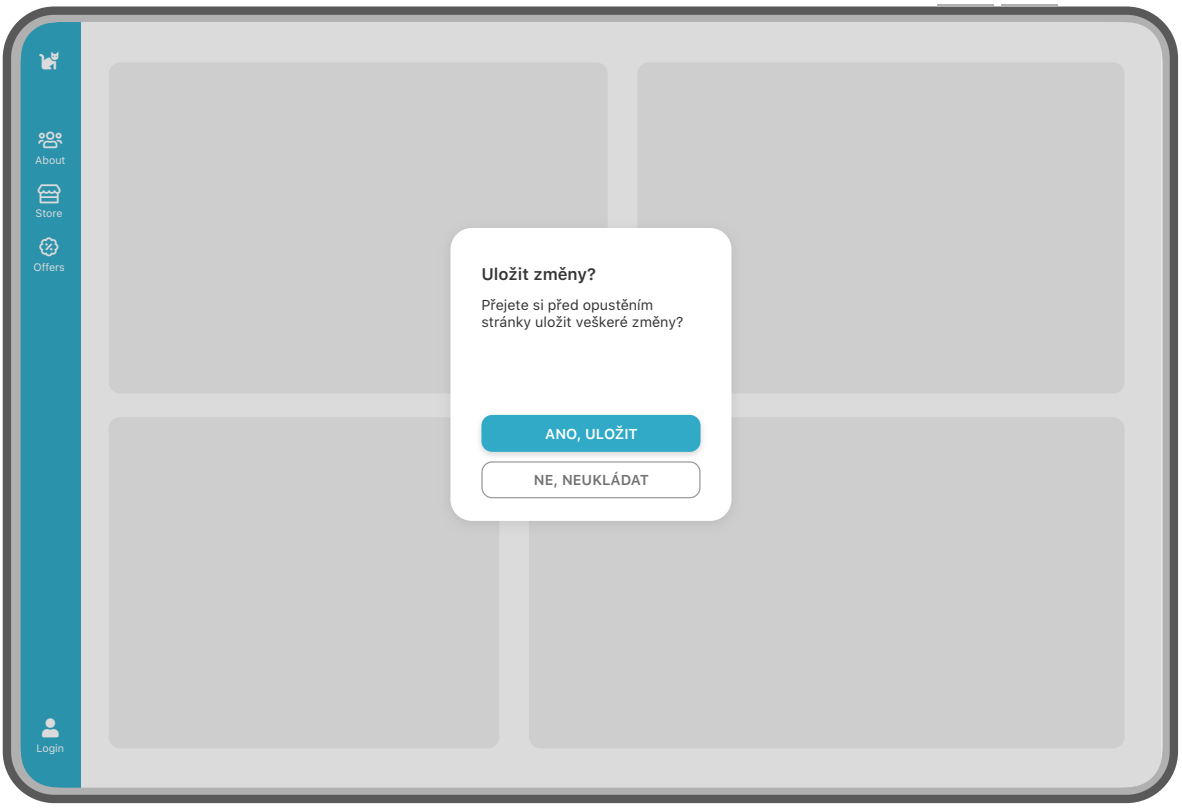
Obrázek 94: Lineární a kruhová varianta progress baru.

Podobně jako tab bar je drop-down list úsporný s ohledem na prostor na obrazovce a poskytuje prevenci zadávání špatných hodnot. Proto je nutné dbát na výstižný a krátký popis obsahu drop-down. Pokud je ruční zadání hodnoty rychlejší než její výběr, je vhodnější použít textový vstup.
Obsah bychom nikdy neměli dynamicky měnit (především jedná-li se o navigaci). Uživatel je nucen si položky po přečtení zapamatovat, což je náročné pokud je opakovaně měníme. Pokud je nutné do obsahu zasahovat, můžeme jeho položky přepnout do stavu „disabled“.
Obdobně jako tlačítko se může drop-down list vyskytovat v několika stavech: default, focus, disabled.

- **default** – Výchozí stav.
- **focus** – Drop-down list po označení (například klávesnicí – důležité pro přístupnost).
- **disabled** – Drop-down list z kterého není umožněn výběr.

Dialog

Dialog slouží k upozornění uživatele. Vyskytuje se jak v modální (uživatel je nucen interagovat, zbytek aplikace je zablokován) tak nemodální variantě. Může obsahovat akce nebo pouze informovat (důležitost informace je vhodné rozlišovat).
Hlavním problémem informačních dialogů může být jejich četnost. Jejich množství by mělo být redukováno na nutné minimum. Při destruktivních akcích je však vždy vhodné uživatele upozornit.



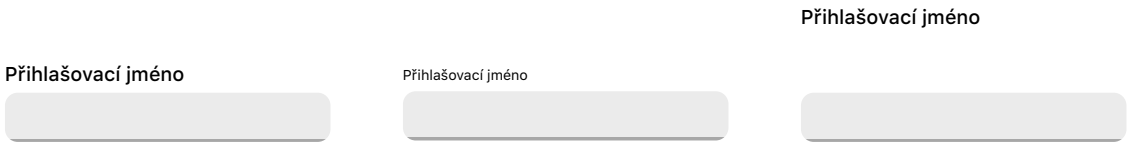
Obrázek 95: Modální dialog.

Uživatelský vstup

V následující sekci se budeme věnovat prvkům umožňující zadávání uživatelského vstupu, ať už se jedná o text nebo výběr z hodnot.

Label

Label neboli *označení/popis* je jedním z nejzákladnějších prvků uživatelského rozhraní (již jsme ho bez explicitního zmínění používali). Obsah popisu nelze zpravidla upravovat a jeho délka by měla být omezena.
Klíčovým výběrem je text samotný a jeho vhodné umístění (whitespace a velikost). Text by měl být krátký a dostatečně výstižný. Velikost textu by měla zaručovat pohodlnou čitelnost, vzdálenost od ostatních prvků pak naznačovat jaký prvek je pojmenován.



Obrázek 96: Popis vlevo má ideální velikost a mezeru od ostatních prvků; prostřední popis je příliš malý; popis vpravo má sice správnou velikost, je však příliš vzdálený od popisovaného prvku.

Text input

Text input neboli *textový vstup* slouží k zadávání textů. Inspirovaný formuláři v papírové podobě. Nalezneme ho nejčastěji ve formuláři nebo dialogovém okně.
Viditelnost a rozeznatelnost je prioritou, nejčastěji volíme obdélníkový tvar. Může obsahovat další prvky jako je například ikona nebo akci pro smazání obsahu textového pole.
Textová pole můžeme rozdělit podle počtu řádků, který vyplněný text zabírá. Nejčastěji se jedná o jednořádkové, dvojřádkové nebo víceřádkové textové vstupy (často nazvané *text area*).
Uživatelskou přívětivost zvýšíme specifikací konkrétního typu vstupu (například jedná-li se o částku v CZK měně nebo e-mail adresu) a jeho následnou validací.

Obrázek 97: Různé varianty textového vstupu.

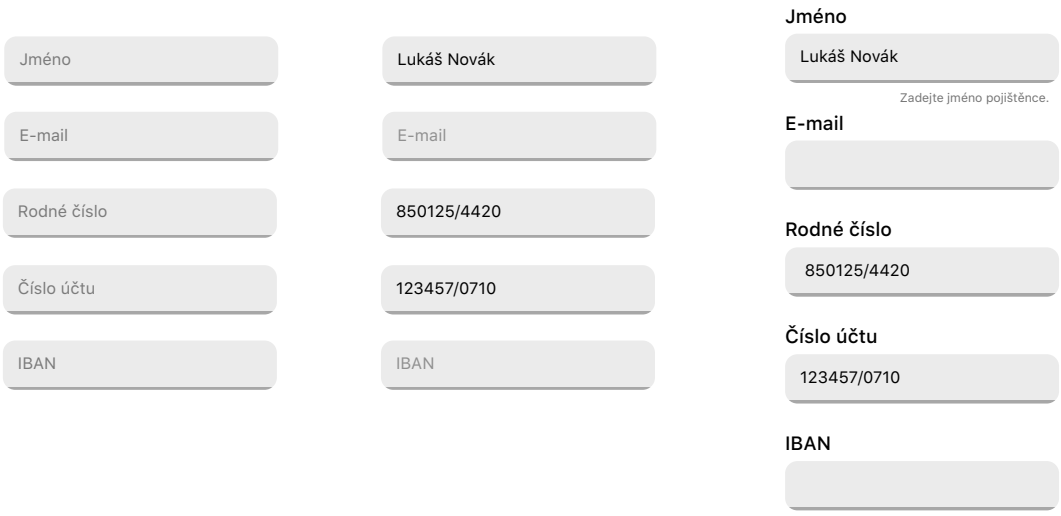
Vyplnění textového vstupu může být povinné, tuto povinnost je dů-

ležité dostatečně výrazně indikovat. V případě, že uživatel povinný textový vstup přehlédne je kritické ostatní vyplněné údaje po neúspěšném odeslání neodstraňovat.
V případě mobilních telefonů je žádoucí zobrazovat vhodný typ klávesnice (pro zadání čísla numerickou) a v maximální míře podporovat automatické formátování a doplňování vstupu.
Textový vstup se může vyskytovat v několika stavech: default, focus, disabled a error.

- **default** – Výchozí stav.
- **focus** – Textový vstup po označení (například klávesnicí – důležité pro přístupnost).
- **disabled** – Textový vstup do kterého nelze zadat vstup.
- **error** – Stav označující chybně vyplněný vstup.

Použití placeholderu

Významnou problematikou textového vstupu je použití *placeholderu* neboli *zástupného textu*. Zástupný text se v textovém vstupu vyskytuje do té doby, než je zadán konkrétní vstup. Často je nevhodně použit, jako popis hodnoty, kterou má uživatel do textového vstupu vložit.



Obrázek 98: První dvě situace demonstrují problém s použitím placeholderu; poslední varianta demonstruje správně navržený formulář.

Jelikož zástupný text po vyplnění hodnoty zmizí, tak tímto použitím vytváříme hned několik problémů:

1. Zatěžujeme zbytečně uživatelovu krátkodobou pamět.
2. Uživatel není schopen ověřit správnost vstupu před odesláním formuláře.
3. Pokud po odeslání formuláře nastane chyba, uživatel není schopen chybu opravit – neví jaká hodnota je očekávána.
4. Zástupný text může zmizet již při umístění kurzoru, to je obtěžující pro uživatele pohybující se pomoci klávesnice.
5. Textové vstupy s vyplněným textem jsou obecně méně nápadné, než textové vstupy prázdné (lákají k vyplnění).
6. Uživatelé si pletou předvyplněné údaje se zástupným textem.

Je tedy vhodné placeholder nepoužívat vůbec, případně v kombinaci s popisky. V případě, že potřebujeme k textovému vstupu přidat poznámku, je vhodnější volit variantu menšího textu pod textovým vstupem.

Picker

Picker neboli *výběr (barvy/data)* je speciální forma drop-down listu sloužící k výběru z několika hodnot (například barvy, datumu, času). Obecně je vždy rovněž dobré umožnit prímý uživatelův vstup (zadání hodnoty).

Date picker Nejčastěji používanou variantou je *date picker* sloužící k výběru datumu nebo jeho rozmezí. S ohledem na uživatelskou přívětivost je ho vhodné používat pouze pro výběr data v blízkém časovém okolí (nikoli třeba datum narození).
V případě mobilních telefonů uživatel většinou odděleně vybere den, měsíc a rok. Na větších obrazovkách by však neměl být výběr rozdělen na jednotlivé části a uživatel by měl vybrat konkrétní datum v jednom kroku.

Obrázek 99: Vlevo správně navržený date picker pro velká zařízení; vpravo nesprávně rozdělený na den, měsíc, rok.

Rovněž je nutné brát ohled na formát data. Názvy měsíců je vhodné uvádět slovy, které uživatel jednodušeji rozezná.

Switch

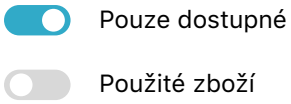
„Every morning, I wake up, pour water into my tea kettle, and flip the switch on. Once the water is boiling, I turn the kettle off and pour myself a cup.“

Alita Joyce,

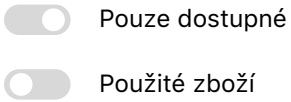
Obrázek 100: Přepínač na horkovarné konvici.

Význam prvku *switch* neboli *přepínače* nejlépe pochopíme na věcech v našem okolí. Spousta zařízení má dvě polohy – zapnuto/vypnuto.
Efekt přepínače nastává okamžitě, pro jeho aktivaci není vyžadováno žádné potvrzovací tlačítko. Jeho použití je tedy častější mimo formuláře (například v nastavení aplikace nebo přepínání filtrů seznamu položek).
U přepínače je nutné jasně indikovat polohu zapnuto/vypnuto, nejlépe jejich přepnutí zdůraznit animací. Musíme si však dát pozor na sémantiku barev (například červená barva je pro indikaci polohy zapnuto nevhodná).
Přepínač se může vyskytovat v několika stavech: default, hover, focus, pressed a disabled.

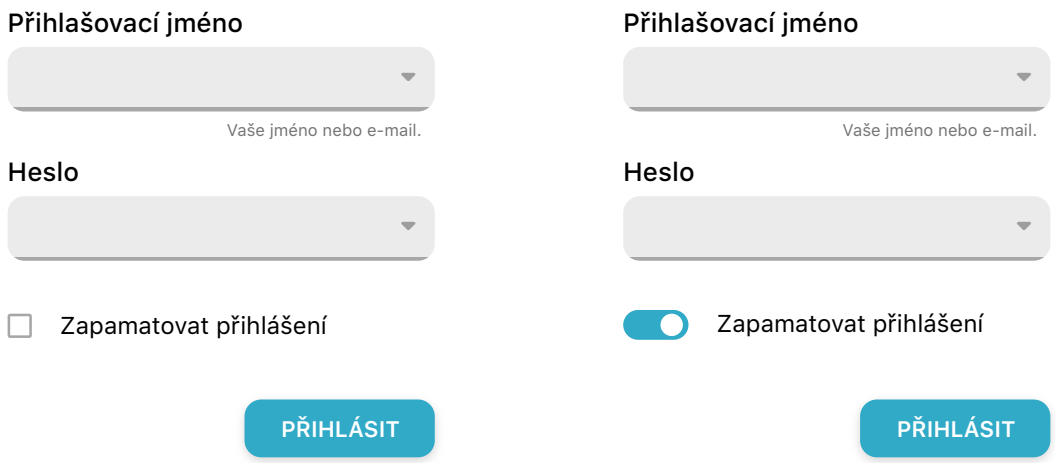
Obrázek 101: Switch, prvek uživatelského rozhraní.



Obrázek 103: prvek switch s absencí barevné indikace.



- **default** – Výchozí stav.
- **hover** – Zvýraznění přepínače po najetí kurzoru.
- **focus** – Přepínač po označení klávesnící (například klávesnicí – důležité pro přístupnost).
- **pressed** – Stav přepínače při zmáčknutí (důležité pro dotykové obrazovky).
- **disabled** – Stav deaktivovaného přepínače – nelze zapnout/-vypnout.

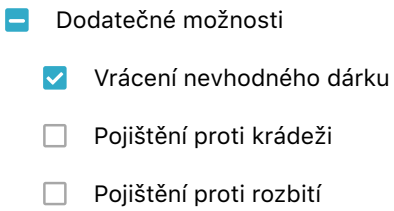


Obrázek 102: Vlevo správně použití checkboxu; vpravo špatně použitý switch.

Checkbox

Checkbox neboli *zaškrtávací pole* je podobně jako textový vstup inspirován papírovými formuláři. Slouží k výběru libovolného počtu hodnot, případně přepíná stav zapnuto/vypnuto podobně jako switch. Narozdíl od switche však checkbox vyžaduje potvrzovací tlačítko (například odeslání formuláře).
Vždy je nutné volit výstižný a ideálně kratší popis. Výchozí hodnota by měla být vždy vhodně zvolena. Předcházíme tím situaci, kdy je uživatel nucen zbytečně označovat či odznačovat větší množství polí.
Zaškrtávací pole je možné umístit do skupiny. Hlavní checkbox poté označuje/odznačuje veškeré podřízené checkboxy. Uživateli je však stále umožněno vybrat libovolnou kombinaci možností.
Checkbox se může vyskytovat v několika stavech: default, hover, focus, pressed a disabled.

- **default** – Výchozí stav.
- **hover** – Zvýraznění checkboxu po najetí kurzoru.
- **focus** – Checkbox po označení klávesnicí (například klávesnicí – důležité pro přístupnost).
- **pressed** – Stav checkboxu při zmáčknutí (důležité pro dotykové obrazovky).



Obrázek 104: Checkboxy umístěné do skupiny.

- **disabled** – Stav deaktivovaného přepínače – nelze zaškrtnout/odškrtnout.

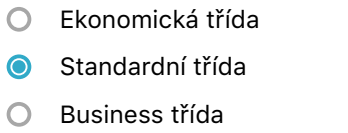
Úkol 11
Na obrázku 105 vidíme dvojci checkboxů, poznáte v čem je (záměrná) chyba? Zaměřte se na jejich popis.

Obrázek 105: Ukázka (záměrně) nevhodných popisků u checkboxů.

Radio button

Radio button neboli *radiový přepínač* je inspirovaný přepínači na starých radiich. V jeden okamžik je možné vybrat právě jednu hodnotu. Tato vlastnost odlišuje radio button od checkboxu.
Kruhovým tvarem je radio button věrný své předloze. Indikace vybrané možnosti realizujeme zabarveným středem.
Radio button je z hlediska prostoru méně úsporný než checkbox/switch (pro realizaci přepínání zapnuto/vypnuto potřebujeme rádiové přepínače). V případě menšího množství hodnot je vhodné preferovat radio button nad drop-down listem, uživatel tak může vidět celý seznam najednou.
Radio button se může vyskytovat v několika stavech: default, hover, focus, pressed a disabled.

Obrázek 106: Prvek radio button pro tři stavy.



- **default** – Výchozí stav.
- **hover** – Zvýraznění po najetí kurzoru.
- **focus** – Radio button po označení klávesnicí (například klávesnicí – důležité pro přístupnost).
- **pressed** – Stav radio button při zmáčknutí (důležité pro dotykové obrazovky).
- **disabled** – Stav deaktivovaného radio button – nelze mezi nimi přepínat.

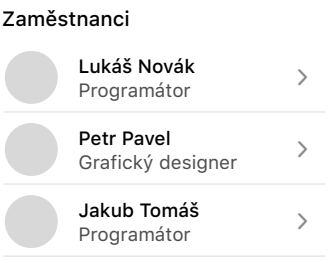
Složené prvky

V následující sekci se budeme věnovat komplexnějším prvkům uživatelského rozhraní, ty jsou většinou složeny z několika základních prvků, které jsme si představili v předchozích sekcích.

List

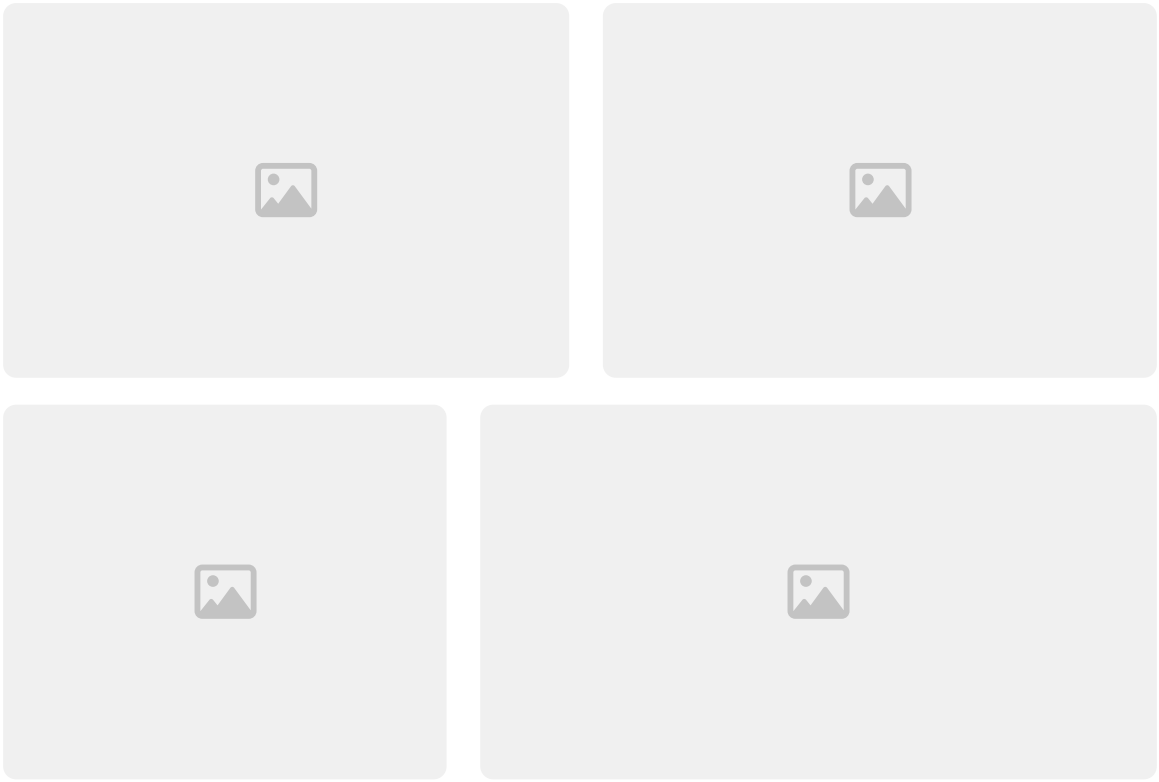
List neboli *seznam* umožňuje uspořádat text a ostatní prvky uživatelského rozhraní do lineárního seznamu. Lineární uspořádání uživatelům usnadňuje rychlé skenování obsahu.
Seznamy rozdělujeme na *jednořádkové*, *dvojřádkové* a *víceřádkové*. Obsah jednotlivých buněk (řádků) je vhodné doplnit o vizuální prvky, které usnadní rozlišení jednotlivých prvků seznamu. Seznamy obsahující holý text mohou působit příliš jednolitě. Jednotlivé prvky seznamu mohou být odděleny separátorem (většinou tenká linka).
V rámci buněk seznamu rovněž můžeme používat primární a sekundární akce (například navigaci na obrazovku s detailními informacemi).

Obrázek 107: Dvojřádkový seznam.



Collections

Collections neboli *kolekce* je nelineární kolekce libovolných prvků uživatelského rozhraní, nejčastěji obrázků nebo karet.

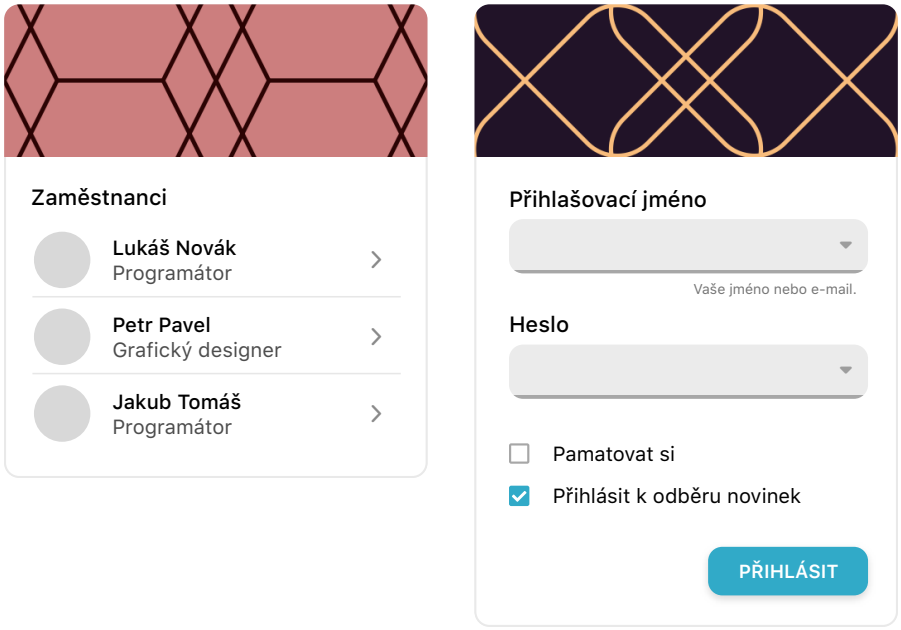


Obrázek 108: Kolekce obrázků

V případě kolekce je skenování obsahu náročnější než u lineárního seznamu. Menší přehlednost je však vyvážena lepší úsporou místa (především na větších obrazovkách).

Card

Card neboli *karta* je složený prvek inspirovaný reálnými kartami. Obsahuje ucelené informace na jednom místě, často spojené s odkazem pro zobrazení detailnějších informací, případě navigací na jiné obrazovky aplikace.
Hlavní výhodou je jejich flexibilní layout, díky kterému můžeme karty jednoduše umisťovat do kolekcí a efektivně využívat prostor obrazovky.



Obrázek 109: Dva příklady použití karet.

Shrnutí

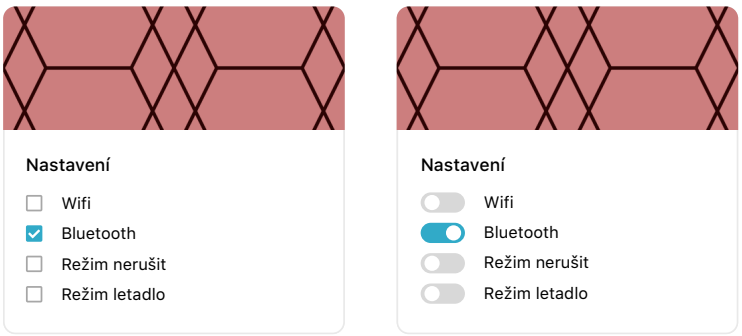
V této kapitole jsme si ukázali výčet základních stavebních kamenů uživatelského rozhraní. Při návrhu aplikace je vždy nutné respektovat pravidla cílové platformy, s tím je často spojena omezená nabídka prvků uživatelského rozhraní. Zároveň je dobré místy aplikace oživovat individuálním přístupem. Aplikace navržené striktně dle pravidel platformy mohou působit až příliš stejnorodě.

Úkoly

Úkol 12
Rozhodněte, které varianta UI na obrázku 110 využívá správně zvolené prvky. Vysvětlete proč.

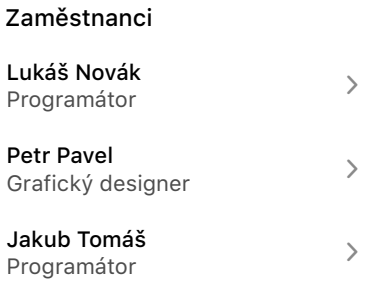
Obrázek 110: Která ze dvou situací je správně a proč?

Úkol 13
Rozhodněte, které varianta UI na obrázku 111 využívá správně zvolené prvky. Vysvětlete proč.



Obrázek 111: Která ze dvou situací je správně a proč?

Úkol 14
Viditelnost seznamu na obrázku 112 je možné vylepšit. Jak by jste seznam modifikovali?



Obrázek 112: Jak můžeme vylepšit viditelnost tohoto seznamu?

Literatura

[1] Wilbert O. Galitz. *The Essential Guide to User Interface Design: An Introduction to GUI Design Principles and Techniques*. Wiley; 3 edition, 2007.

[2] Google. Material design guidelines. `https://material.io/design`.

[3] Google. Understanding navigation. `https://material.io/design/navigation/understanding-navigation.html`.

[4] Nielsen Norman Group. The aesthetic-usability effect. `https://www.nngroup.com/articles/aesthetic-usability-effect/`.

[5] Nielsen Norman Group. Flat ui elements attract less attention and cause uncertainty. `https://www.nngroup.com/articles/flat-ui-less-attention-cause-uncertainty/`.

[6] Nielsen Norman Group. Serif vs. sans-serif fonts for hd screens. `https://www.nngroup.com/articles/serif-vs-sans-serif-fonts-hd-screens`.

[7] Daniel Kahneman. *Thinking, Fast and Slow*. Farrar, Straus and Giroux, 1st edition edition, 2013.

[8] Steve Krug. *Don't Make Me Think, Revisited: A Common Sense Approach to Web Usability*. New Riders; 3 edition, 2014.

[9] Everett N. McKay. *UI is Communication: How to Design Intuitive, User Centered Interfaces by Focusing on Effective Communication*. Morgan Kaufmann; 1 edition, 2013.

[10] George A. Miller. The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological Review*, 1994.

[11] Donald A. Norman. *The Design of Everyday Things*. Basic Books, revised and expanded edition edition, 2013.

[12] Shan Shen. Make sense of rounded corners on buttons. https://uxdesign.cc/make-sense-of-rounded-corners-on-buttons-dfc8e13ea7f7/.

[13] Jenifer Tidwell. *Designing Interfaces: Patterns for Effective Interaction Design*. O'Reilly Media; Second edition, 2011.

[14] Robin Williams. *The Non-Designer's Design Book*. Peachpit Press; 4 edition, 2014.

[15] Dennis L. Byrnes Wingfield, Arthur. *The psychology of human memory*. New York: Academic Press, 1981.

[16] Jon Yablonski. Laws of ux. https://lawsofux.com.