SKAISi FE arendusreeglid

FE arendusreeglid

Lühendid

FE – *Front-end*
BE – Back-end
FF – Mozilla Firefox
IE – Internet Explorer
UI – *User interface* (kasutajaliides)
SEO – Search engine optimization

Eesmärk ja sihtgrupp
FE arendusreeglid on mõeldud kõigile FE ja BE arendajatele. Reeglite järgimine on kohustuslik.

Veebilehitsejate tugi

Laua ja sülearvutid

- A klass: IE11+, Chrome (viimane versioon), FF (viimane versioon), Microsoft Edge (viimane versioon)

  Lõpptulem ei tohi kujundusest erineda, välja arvatud veebilehitsejate implementatsioonist tulenevad erandid (näide: selectboxi native kujundus on veebilehitsejates omavahel erinev ja see on lubatud)

- B klass: Safari

  Visuaalselt korrektne, aga võib erineda algsest kujundusest.

Mobiilsed seadmed

- IOS + Safari
- Android + Samsung Internet
- Android+ Chrome

Abistavad tehnoloogiad

Iseteenindus peab olema kasutatav järgmiste ekraanilugeritega

- Windows – JAWS
- Windows – NVDA
- OSX – VoiceOver

Ekraanilugerid peavad töötama järgmistes ekraaniluger–veebilehitseja kombinatsioonides

- JAWS+InternetExplorer
- JAWS+GoogleChrome
- NVDA+Firefox
- VoiceOver+Safari

Stiililehed

Peab lähtuma RIA stiiliraamatus kasutatud praktikatele.

Kataloogide stuktuur

Peab lähtuma RIA stiiliraamatus kasutatud praktikatele.

Style-Guide Driven Development ja Living Style-Guide

Selleks, et võimalikult palju front-end komponentidest oleks dokumenteeritud ja lihtsasti leitavad tuleb rakendada style-guide driven developmenti. Üldiselt tähendab see – kui tekib vajadus uue komponendi järgi, mida veel süsteemis kasutusel ei ole siis esmalt dokumenteeritakse see stiiliraamatus. Kindlasti peavad dokumenteeritud saama komponendid, mida kasutatakse rohkem, kui ühes kohas. Living Style-Guide kontsept tähendab seda, et infosüsteemi kasutajaliidese dokumentatsioon ja rakenduse lähtekood on alati sünkroonis ja kõik ajas toimuvad muudatused saavad dokumenteeritud.

Dokumentatsioonis peab sisaldama

- Komponendid ja nende erilahendused. N: nuppude puhul nende suuruste ja värvi variatsioonid.
    - Näidised kuidas komponenti rakenduses välja kutsuda
    - Komponendiga seotud HTML ja CSS classid
    - Sisend parameetrid
    - Meetodid ja funktsioonid

Re-usable front-end

1. Enne kodeerimist tutvuda olemasolevate komponentidega.
2. Võimalusel kasutada olemasolevaid elemente.
3. Kujunduses olevate väikeste erinevuste puhul pakkuda välja ühtlustamise võimalusi ja lahendusi.
4. Kasutada üldiseid classi nimetusi ja arvestada kodeerimisel, et antud stiile oleks võimalik kasutada läbivalt. Erandite puhul kasutada eraldi classi nimetust.
5. Kodeerimisel arvestada, et uue elemendi muutmine ja haldamine oleks võimalikult lihtne. Näiteks arvestada keelsusega. Võimalusel vältida keelte jaoks erandite tegemist.

CSSi haldamine

Selleks, et tagada kasutajaliidese ühtne väljanägemine ja vähendada sarnaste stiilide topelt kirjutamist on stiilide haldus reglementeeritud. Seni kuni see ei muutu arenduses pudelikaelaks on stiilidel üks haldaja ja vastutaja, kes on kursis kõigi muudatustega. Tema kohustus on üle vaadata teiste poolt kirjutatud CSS ja vastavalt vajadusele muuta või saata tagasi arendajale parandamiseks. Arendusmeeskonna ja mahtude kasvades tuleb stiilide haldamise rolli laiendada laiemale meeskonnale.

Illustreerivad pildid

- Illustreerivad pildid peavad olema lisatud CSSi abiga.
- Pildid on optimeeritud
- Eelistada illustreerivate piltide kuvamist läbi CSSi.
- HTMLis lingitud piltide nimetamisel arvestada SEOd ja ühtset stiili.
- CSSi pildid nimetada sarnase loogikaga. Näiteks: Ikoonide alguses on icon_, taustapiltidel bg_ logodel logo_, nimekirjadel ul_ ja komponentide spetiifiliste piltidel puhul selle classi nimetus. Kui kasutatakse CSS raamistikku siis lähtuda raamistikus kehtestatud reeglitest
- Pildi nimed on läbivalt inglise keeles ja semantiliste nimetusega. Ei tohi kirjeldada pildil olevat värvi.

Semantika

Front-end kood peab vastama semantika reeglitele, välja arvatud juhul, kui kokkulepitud kasutatav raamistik ei võimalda seda saavutada. Kodeerimisel tuleb arvestada *responsive (kohandub iseenesest ümber erinevatele ekraanisuurustele)* versioonile üleminekuga.

Olulisemad punktid

- Classi nimed on üldised või komponendi spetsiifilised
- Kasutada semantiliselt korrektseid HTMLi elemente.
    - Näiteks nimekirjade puhul peab olema kasutatud UL või OL tagi vastavalt sisule.
    - BR ei tohi kasutada paigutuse loomiseks. Võib kasutada otstrabeliselt tekstide sees
    - Tabeleid tohib kasutada ainult tabelite jaoks, mitte üldise lehe struktuuri loomiseks.
    - Inline elemendi sisse ei tohi panna block elemente.
    - Paragraafide sisse ei tohi kirjutada DIVe.
- Vältida tühjade HTMLi elementide kasutamist.

HTMLi reeglid

- HTML peab valideeruma kasutades https://validator.w3.org HTML valideerimisvahendit.
    - Arvestada, et eristatakse Angular– ja HTML spetsiifilisi valideerimise tulemusi. – Eesmärk on tagada korrektne ja standardile vastav HTML kood aga ignoreerida Angular raamistikust tulenevaid HTML valideerimise veateateid.
- Inline CSSi ei tohi kirjutada ja kasutada.
- *Cellpadding*'uid ja *cellspacing*'uid ei tohi kasutada HTMLis.
- Võimalusel kirjutada optimeeritud koodi ja vältida üleliigseid elemente ja defineeringuid.
- Kasutada läbivalt ühtset stiili.
    - Jutumärkideks on topelt jutumärgid
        - näide class="style01"
    - Arenduse failid on ühe tabi abil trepitud.
- Arvestada SEO soovitusi
    - Vajadusel illustreeritavatel piltidel korrektsed alt väärtused.
    - Meta tagid vastavalt kaasaegsetele nõudmistele.
- Piltide või ikoonide puhul, mis on kasutusel linkidena, kasutada title tagi. Title sisu peab kasutajale andma informatsiooni, et kuhu see link kasutaja suunab.
    - Näide logo title väärtus on "Avalehele"
- HTMLis vältida ja eemaldada üleliigsed (näiteks arendajate kommentaarid) kommentaarid.

CSSi reeglid

Kehtib FE arendaja poolt loodud stiililehtede kohta. Ei kehti juhul, kui kokkulepitud kasutatav raamistik ei võimalda seda saavutada.

Iseteenindus hakkab kasutama RIA stiiliraamatut, mis on ehitatud Bootstrap4 CSS raamistikule – Iseteeninduse stiilid peaksid jälgima Bootstrap css juhiseid http://codeguide.co/#css. Allikas https://github.com/twbs/bootstrap/blob/v4.0.0/.github/CONTRIBUTING.md#code-guidelines

- CSS peab valideeruma kasutades https://jigsaw.w3.org/css-validator/ CSS valideerimisvahendit.
    - CSS valideerimisel võetakse aluseks profiil CSS level 3 + SVG, brauseri tootjate spetsiifiliste (vendor prefix) css reegleid käsitletakse kui hoiatusi ja nende kasutamine on aktsepteeritav. Kui tekib vajadus vanemate brauserite toetamiseks kasutada mitte valideeruvat CSS-i siis selles lepitakse eraldi kokku.
- CSSi treppimine
    - Selektori ja loogeliste sulgude vahele üks tühik.
    - Üks tab enne atribuuti (laiuselt võrdne nelja tühikuga).
    - Selektorid ja atribuudid on eraldi ridadel.
    - Näide

```
ul*{
****margin:*0;
}
```

- Selectori blokkide vahel tühjad read eraldamaks erinevaid elemente aga üldiselt ilma tühja reata.

```
h1 {
    font-size: 34px;
}
h2 {
    font-size: 24px;
}
a {
    color: #08c;
}
a:focus,
a:hover {
    color: #ff6;
}
```

- Esimese taseme ja teise taseme kommentaari ees ja järgi üks tühi rida.
- CSSis blokkide järjekord:
    - normalize/reset, global
    - common
    - vastavalt struktuurile header, content ...
        - media queries hoida komponendi lähedal
    - Plugina CSSid, kui on tegemist pluginaga, mida kasutatakse samade stiilidega erinevates asukohtades, siis antud CSS lisada eraldi faili.

- CSS Atribuutide järjekord:

```
.declaration-order {
  /* Positioning */
  position: absolute;
  top: 0;
  right: 0;
  bottom: 0;
  left: 0;
  z-index: 100;

  /* Box-model */
  display: block;
  float: right;
  width: 100px;
  height: 100px;

  /* Typography */
  font: normal 13px "Helvetica Neue", sans-serif;
  line-height: 1.5;
  color: #333;
  text-align: center;

  /* Visual */
  background-color: #f5f5f5;
  border: 1px solid #e5e5e5;
  border-radius: 3px;

  /* Misc */
  opacity: 1;
}
```

- Ühikud
  - Eelistatult läbivalt kasutada rem ühikuid. https://getbootstrap.com/docs/4.1/migration/#global-changes
  - 0 väärtuse korral ühik pole vajalik. 0em == 0px == 0
- Class' ja ID nimetused läbivalt väikeste tähtedega, semantilised ja eralduseks kasutatud sidekriipsu.
  - Näide: data-table-outer
- Classide nimetamisel kasutada läbivalt inglise keelt.
- ID kasutamine stiilide jaoks on välistatud
- CSSis on kaks taset kommentaare.
  1. esimene tase
     /**
     =Kommentaar
     */
  2. teine tase
     /** =Kommentaar */
- Kommentaarid läbivalt inglise keeles. Vajadusel rakenduse spetiifilisi mõisteid kasutada eesti keeles.
- Välja kommenteeritud koodi ei tohi jätta alles.
- Margin ja paddingute korrektne kasutamine. Paddingut kasutada antud elemendi sisse nn õhu jätmiseks ja marginit väljapoole.
- Tekstide joondamisel vältida line-heighti abil joondamist ja eelistada paddingut.
- CSS kirjutada ainult selleks mõeldud faili. Inline CSSi ja HTMLi failis eraldi <style> kasutada ei tohiks. Võimalusel vältida ka visuaalsete stiilide defineerimist JS failides.
- Pildi URLi lisamisel jutumärke ei ole vaja kasutada.
- Vältida veebilehitsejaspetiifilisi häkke.
- Kodeerimisel arvestada WCAG 2.1 AA taseme ligipääsetavuse nõuetega
  - http://www.w3.org/WAI/WCAG20/quickref/
  - http://kristjankure.com/

Eeltöötlus (preprocessor) keeled

Kasutada SASS preprocessor keelt SCSS süntaksiga.

Selektorite nestimine

Selektorite *nestimisel* tuleb piirduda kolme tasemega. Üle kolme taseme *nestimine* teeb koodi lugemise liialt keeruliseks

```
.element {
  .element-child {
    p {
      //Siit sügavamale ära mine
    }
  }
}
```

Optimeerimine FE tehnikate abil

- https://developer.mozilla.org/en-US/docs/Web/Guide/CSS/Writing_efficient_CSS
    - tbody#select-01 => #select-01
    - #navigation li a => #nav a
    - tr.tr-name td.td-name => .td-name
- HTMLis olevatele piltide määrata kõrgus ja laius, kui on võimalik. Eesmärk vähendada re-flow'ed.
- Gradientide, ümarate nurkade jne puhul kasutada CSS3 ning vanematel veebilehitsejatel peab kujundus välja nägema visuaalselt korrektne aga ei pea olema visuaalselt identne. Kontrollida veebilehitseja võimalusi siit http://caniuse.com/.
- Kui on võimalik illustreerivad pildid nt nooled, bulletid asendada CSS3 lahendusega, siis seda võimalust kasutada. Kui vajadus tühja elementi kasutada, siis selle asemel kasutada CSSis before'i võimalust. Selle juures tuleks arvestada veebilehitsejate tuge ja kasutajamugavust mõistlikuse piirides. Informatsioon peab olema arusaadav ja kättesaadav ka B klassi veebilehitseja toele.
- Reflow ja repaint'iga arvestamine.
- CSSi reseti asemel kasutada normalize.
- Värvid kirjutada koodi abil (soovitavalt hex aga vajadusel rgba) ja läbivalt väikeste tähtedega. Võimalusel lühendada kolme tähelisteks.
    - #ffffff => #fff
    - #ff88aa => #f8a, #ff0000 => #f00
- Võimalusel CSSi väärtused tõsta kokku ühele reale.
    - Border-top, border-left, border-bottom, border-right => border
- Important kasutamist CSSi failis vältida.
- Normalize/reseti puhul ei tohiks olla defineeritud elemente, mida praktikas ei kasutata. Uue projekti alguses tuleks reset/normalize optimeerida ja vajadusel jooksvalt juurde lisada elemente.
- Võimalusel ja vajadusel kasutada sprite tehnikat.
- Pluginatega kaasa tulevat CSSi peab ülevaatama, optimeerima ja järjekorda uuendama.

Kujundus

- Linkidele, nuppudele alati korrektsed :hover, :focus ja :active staatused määrata.
    - hoveri ja fookuse puhul on eesmärk tuua visuaalselt element rohkem esile ja paremini loetavaks.
    - aktiivse staatuse puhul tuleks jätta mulje, et nupule või lingile on vajutatud. Visuaalselt poolelt näiteks gradient on vastupidine või lingi värv on tagasihoidlikum.
- Arvestada olemasoleva kujundusega ja jälgida sellele sarnast stiili.
- Ühtlustada ja jälgida lõpptulemuse joonduseid.