

Curso Técnico Superior Profissional em Tecnologias e Programação de Sistemas de Informação

Unidade Curricular: Design Hipermédia

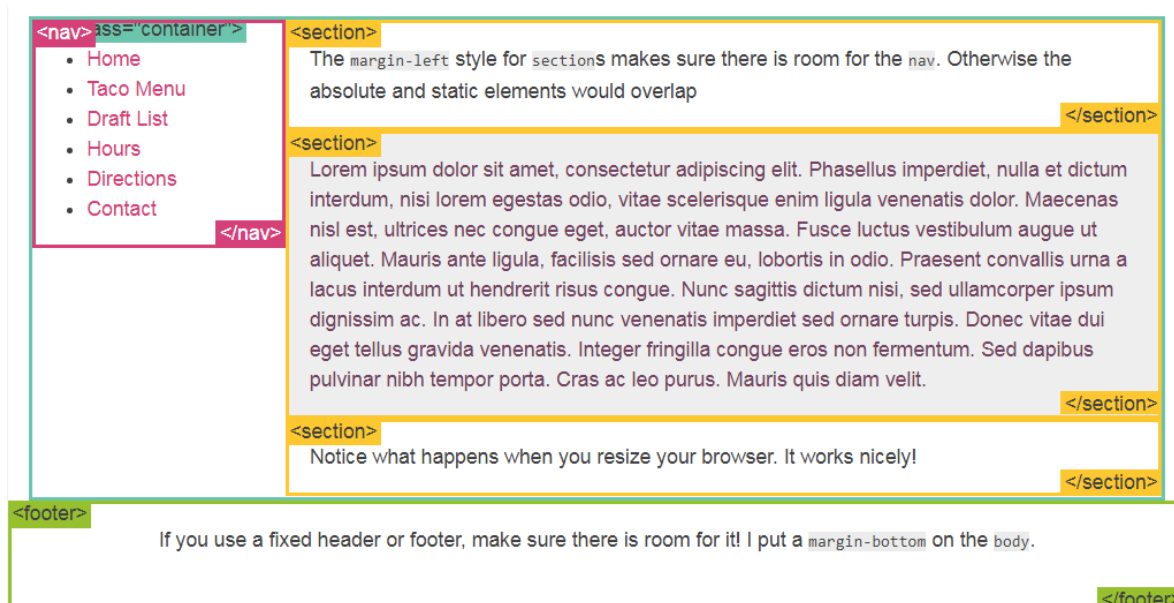
1º Ano/1º Semestre

Docente: Marco Miguel Olival Olim

Data 30/10/2017

ESTE EXERCÍCIO PRETENDE ILUSTRAR AS DIFERENÇAS ENTRE AS DIFERENTES TÉCNICAS PARA CONCEBER LAYOUTS EM CSS

- 1- Considere a seguinte estrutura. Pretende-se recriar esta estrutura usando as diferentes técnicas de Layout estudadas, nomeadamente “Float”, “FlexBox” e “Grid”. Utilize só em último recurso a proposta de implementação do css apresentada em baixo dos enunciados:



```

<nav class="container">
  • Home
  • Taco Menu
  • Draft List
  • Hours
  • Directions
  • Contact
</nav>

<section>
  The margin-left style for sections makes sure there is room for the nav. Otherwise the
  absolute and static elements would overlap
</section>

<section>
  Lorem ipsum dolor sit amet, consectetur adipiscing elit. Phasellus imperdiet, nulla et dictum
  interdum, nisi lorem egestas odio, vitae scelerisque enim ligula venenatis dolor. Maecenas
  nisl est, ultrices nec congue eget, auctor vitae massa. Fusce luctus vestibulum augue ut
  aliquet. Mauris ante ligula, facilisis sed ornare eu, lobortis in odio. Praesent convallis urna a
  lacus interdum ut hendrerit risus congue. Nunc sagittis dictum nisi, sed ullamcorper ipsum
  dignissim ac. In at libero sed nunc venenatis imperdiet sed ornare turpis. Donec vitae dui
  eget tellus gravida venenatis. Integer fringilla congue eros non fermentum. Sed dapibus
  pulvinar nibh tempor porta. Cras ac leo purus. Mauris quis diam velit.
</section>

<section>
  Notice what happens when you resize your browser. It works nicely!
</section>

<footer>
  If you use a fixed header or footer, make sure there is room for it! I put a margin-bottom on the body.
</footer>
  
```

- 2- Para começar o exercício crie um ficheiro html (e css associado) e procure replicar o layout acima apresentado utilizando apenas posicionamentos.

```

.container {
  position: relative;
}
nav {
  position: absolute;
  left: 0px;
  width: 200px;
}
section {
  /* position is static by default */
  margin-left: 200px;
}
footer {
  position: fixed;
  bottom: 0;
  left: 0;
  height: 70px;
  background-color: white;
  width: 100%;
}
body {
  margin-bottom: 120px;
}
  
```

Cofinanciado por:

- 3- Crie agora um segundo conjunto de ficheiros HTML e CSS e utilize *inline-block* para o Layout:

```
nav {
  display: inline-block;
  vertical-align: top;
  width: 25%;
}
.column {
  display: inline-block;
  vertical-align: top;
  width: 75%;
}
```

- 4- Para o terceiro conjunto utilize *floats*. Abaixo é também apresentada uma solução para dispositivos móveis utilizando media queries:

```
@media screen and (min-width:600px) {
  nav {
    float: left;
    width: 25%;
  }
  section {
    margin-left: 25%;
  }
}
@media screen and (max-width:599px) {
  nav li {
    display: inline;
  }
}
```

- 5- Para o quarto conjunto utilize FlexBox. Neste exemplo foi utilizado uma classe flex-column num div que envolve todas as secções.

```
.container {
  display: -webkit-flex;
  display: flex;
}
nav {
  width: 200px;
}
.flex-column {
  -webkit-flex: 1;
  flex: 1;
}
```

- 6- Conclua o exercício utilizando a última técnica estudada para criação de layouts – o Grid

```
.container{
  display: grid;
  grid-template-columns: 25% auto;
  grid-template-rows: 1fr 1fr;
}
nav{
  grid-column-start: 1;
  grid-column-end: 2;
  grid-row-start: 1;
  grid-row-end: 3;
}
.grid-column{
  grid-column-start: 2;
  grid-column-end: 3;
  grid-row-start: 1;
  grid-row-end: span 2;
}
```

Cofinanciado por: