Coloque também as informações que o navegador necessita para exibir corretamente a página: as tags <meta>, <link> e <title>. Não esqueça também do DOCTYPE.

Aqui, seria interessante dar destaque para os termos "qualidade", "durabilidade" e "agregam valor", deixando-os em *itálico*, por exemplo. Para fazer isso, usamos a tag <em>:

deixando-os em negrito. Para isso, podemos usar a tag <strong>:

baixar fonts

<https://fonts.google.com/>

Fzer Um menu

<aside>

<h1>João da Silva</h1>

<nav>

<ul>

<li><a href="index.html">Home</a></li>

<li><a href="portfolio.html">Portfolio</a></li>

<li><a href="bio.html">Sobre mim</a></li>

<li><a href="blog.html">Blog</a></li>

<li><a href="contato.html">Contato</a></li>

</ul>

</nav>

</aside>

Podemos controlar o sublinhamento de um texto usando a propriedade text-decoration. Se usarmos o valor none, o texto aparece sem sublinhado; se usarmos o valor underline, o texto aparece sublinhado.

É possível fazer desenhos e formas bem interessantes apenas usando CSS. Por exemplo, o símbolo universal de pausar (duas barras verticais) pode ser feito com um elemento <div> vazio (ou qualquer outro elemento como <p> ou <header>) com duas bordas grossas, uma à esquerda e outra à direita, e um espaço entre elas, que pode ser a largura do elemento:

div {

border-left: 10px solid red;

border-right: 10px solid red;

width: 5px;

height: 30px;

}

Também é possível desenhar um triângulo apenas usando CSS e bordas. Basta fazer a borda inferior ter uma cor e as bordas vizinhas serem transparentes; a mudança de cor entre as bordas ocorre nas diagonais do elemento, então a borda inferior vai ficar com a aparência de um triângulo, desde que o elemento tenha zero pixels de largura (senão vai parecer um trapézio):

div {

border-bottom: 30px solid red;

border-left: 30px solid transparent;

border-right: 30px solid transparent;

width: 0;

}

Tente fazer um elemento que se parece com uma gravata borboleta e um que seja um triângulo com um ângulo reto.

NADA é OQUE PARECE <<------------------------

Como já discutimos antes, quando desenvolvemos uma página, precisamos levar sempre em conta não só a aparência dos elementos da página, mas também o que eles significam, ou seja, a **semântica** deles.

Quando começamos a levá-la em conta, percebemos que nem sempre o mapeamento entre a aparência de um elemento e o HTML é direto. Considere, por exemplo, o título da página. Atualmente, ele está assim: com letras maiúsculas e minúsculas.



Porém, se olharmos no layout original, aquele que queremos implementar, o título é feito apenas de letras maiúsculas:



A primeira solução que podemos pensar é mudar o HTML para deixar o texto com letras maiúsculas:

<h1>SOBRE MIM</h1>

Para quem *entende* português, o significado não muda, pois ainda é possível identificar as palavras "sobre" e "mim". Mas, para alguém que não entende português, o que é isso? Quando usamos apenas letras maiúsculas, não estamos escrevendo uma palavra, mas sim uma **sigla**. Então, quem não *entende* português vai achar que isso é uma sigla.

Dentre aqueles que não *entendem* português, podemos citar, além das *pessoas* que não falam português, as **máquinas**. Isto é, programas como os leitores de tela e as ferramentas de busca não *entendem* o que nosso texto quer dizer. Isso significa, então, que os programas leitores de tela e as ferramentas de busca ***não entendem*** o que nossa página significa; para eles, tudo são apenas símbolos.

Então como podemos fazer para manter o *significado* original, de palavra, e deixar o *visual* com letras maiúsculas? A própria pergunta já traz a resposta: o HTML é onde definimos o significado e o CSS é onde definimos a aparência. Então, no HTML, mantemos o texto com palavras:

<h1>Sobre mim</h1>

E, no CSS, mudamos a aparência do texto usando a propriedade text-transform, dizendo que queremos que esse texto seja exibido com letras maiúsculas:

h1 {

text-transform: uppercase;

}

## **Nem toda imagem é uma imagem**

Anteriormente, colocamos os ícones das redes sociais na barra lateral usando uma lista para representá-las. Conseguimos o resultado esperado, mas será que a semântica está correta?

Quando pensamos em colocar imagens na página, a primeira ideia que vem à mente é colocar uma tag <img>. Porém, precisamos pensar, como sempre, na sêmantica. Por que essas imagens estão lá? Elas estão lá para representar uma informação visual? Ou estão lá apenas para *representar visualmente* o nome de uma rede social? Se estão lá apenas porque representam **visualmente** o nome da rede social correspondente, deveríamos deixar o lado visual para o CSS, e deixar no HTML o nome da rede social, que é o que queremos passar de significado. Ou seja, nosso HTML ficaria assim:

<ul class="icones-sociais">

<li>

<a href="https://github.com/joaodasilva">

Github

</a>

</li>

<li>

<a href="https://twitter.com/joaodasilva">

Twitter

</a>

</li>

<li>

<a href="https://br.linkedin.com/pub/joão-da-silva/32/4/508">

LinkedIn

</a>

</li>

</ul>

Agora, precisamos fazer a imagem aparecer via CSS e, além disso, fazer o texto *sumir*. Colocar a imagem via CSS é possível graças à propriedade background-image.

.icones-sociais a {

background-image: url(github.png);

}

Fazendo isso, veremos a imagem do Github atrás do texto dos links.

Logo do Github cortado atrás dos textos dos links

Repare que o logo do Github está cortado. Precisamos, então, definir um tamanho para o link, de modo que a imagem inteira apareça no fundo do elemento. Para isso, usamos as propriedades width e height. Porém, para que essas propriedades funcionem com nossos links, é necessário que eles tenham um display que seja inline-block ou block. Vamos usar o valor block, mas poderia ser inline-block também.

.icones-sociais a {

background-image: url(github.png);

width: 40px;

height: 40px;

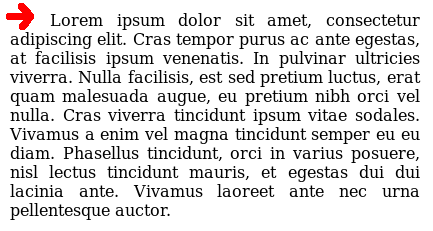
display: block;

}

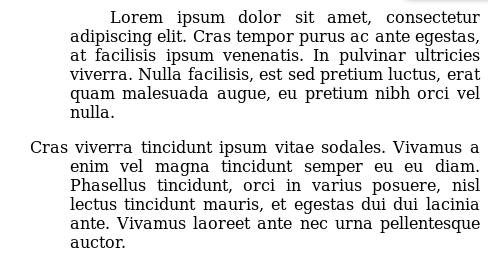
Com isso, já conseguimos ver a imagem inteira, mas o texto continua em cima dela.

Logo do Github completo, mas atrás do texto

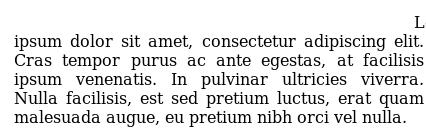
Como remover o texto? No CSS, existe a propriedade text-indent, que permite regular o recuo da primeira linha de um parágrafo.



Quando colocamos um tamanho maior que zero para essa propriedade, a primeira linha começa um pouco *depois* das linhas seguintes. Se colocamos um tamanho *menor que zero*, a primeira linha começa um pouco *antes*.



Se aumentamos muito o valor do text-indent, o texto pode começar fora da página.



Mas, se o texto tiver mais de uma palavra, apenas a primeira palavra ficará para fora da página. Então podemos tentar com um tamanho **bem** menor que zero. No nosso caso, nosso CSS vai ficar parecido com:

.icones-sociais a {

background-image: url(github.png);

width: 40px;

height: 40px;

display: block;

text-indent: -99999px;

}

Com isso, conseguimos o efeito desejado.

Logotipo do Github sem o texto por cima

Conseguimos, assim, substituir um **texto** do HTML por uma **imagem** carregada via CSS. Essa é uma prática comum, tão comum que até tem um nome popular: ***image replacement***.

Só falta arrumar um problema: **todos** os links estão com a imagem do Github. Isso acontece porque, no CSS, estamos selecionando **todos** os links de uma vez. Para selecionar cada link de uma vez, podemos colocar uma classe para cada um:

<ul class="icones-sociais">

<li>

<a href="https://github.com/joaodasilva" class="github">

Github

</a>

</li>

<li>

<a href="https://twitter.com/joaodasilva" class="twitter">

Twitter

</a>

</li>

<li>

<a href="https://br.linkedin.com/pub/joão-da-silva/32/4/508" class="linkedin">

LinkedIn

</a>

</li>

</ul>

No CSS, podemos manter boa parte das regras no seletor .icones-sociais a. Apenas a imagem de fundo muda, então apenas essa declaração precisa ficar na classe específica do link:

.icones-sociais a {

width: 40px;

height: 40px;

display: inline-block;

text-indent: -99999px;

}

.github {

background-image: url(github.png);

}

.twitter {

background-image: url(twitter.png);

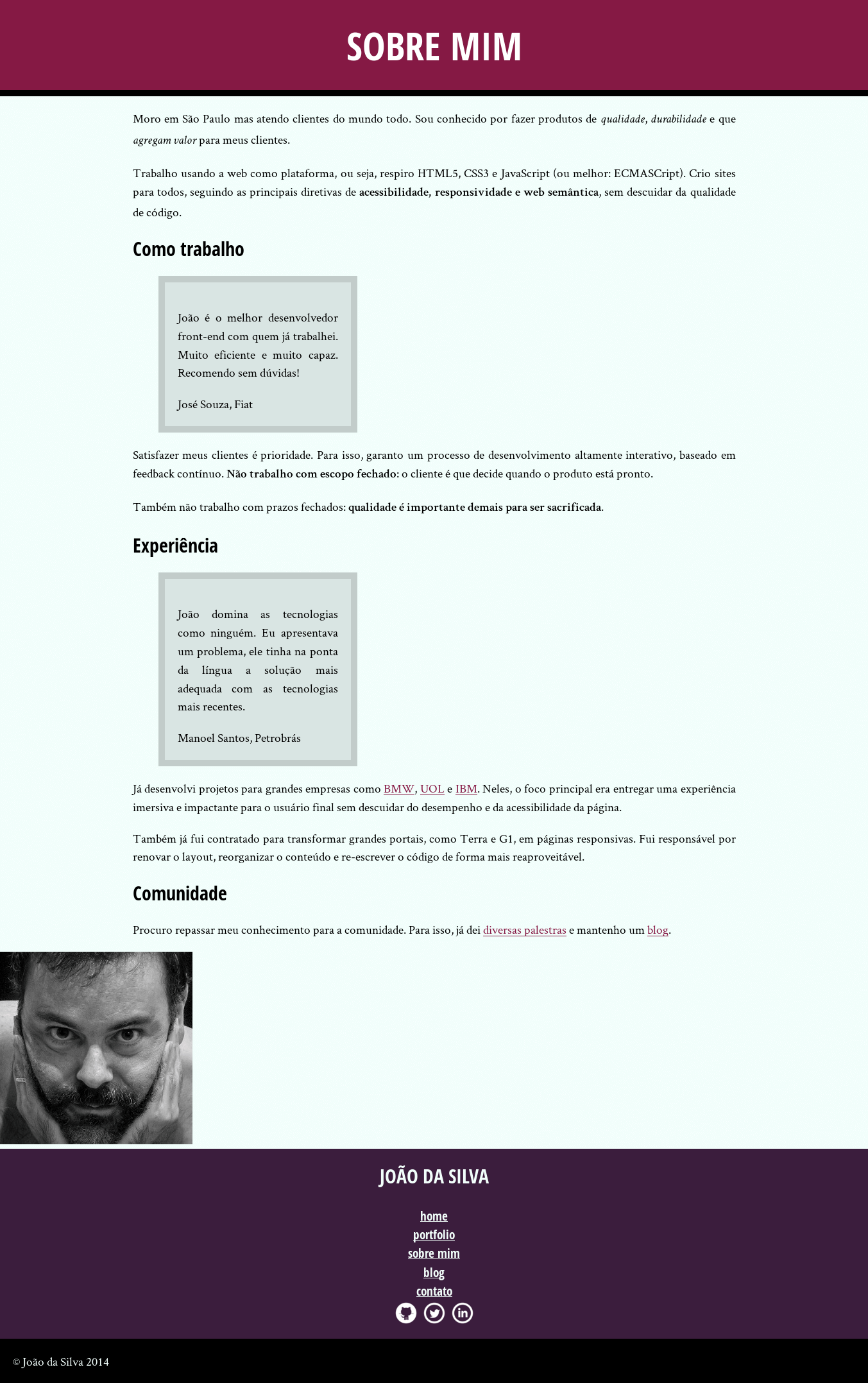
}

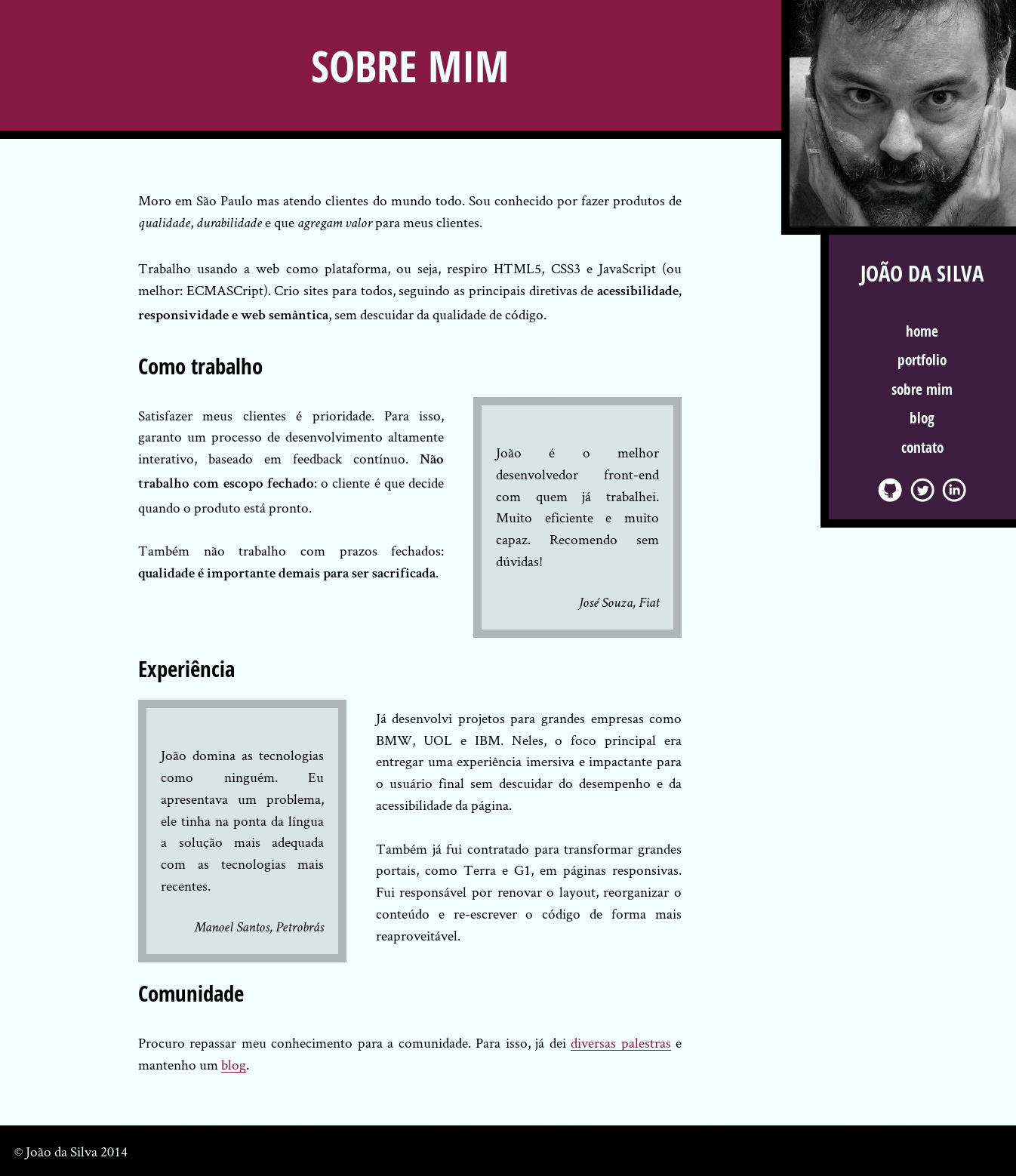
.linkedin {

background-image: url(linkedin.png);

}

Nossa página está quase pronta. Resta apenas colocar a barra lateral, a foto e as citações nos seus devidos lugares.

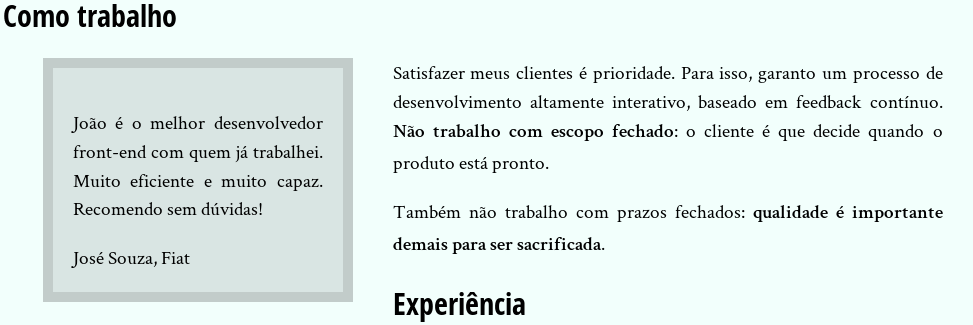




Vamos começar pelas citações. Repare que, no final, cada uma deve ficar de um lado do texto. Vimos que a propriedade display permite colocar um elemento ao lado do outro. Porém, ela não permite mudar a ordem de exibição dos elementos: o que vem primeiro no HTML, vem primeiro no visual.

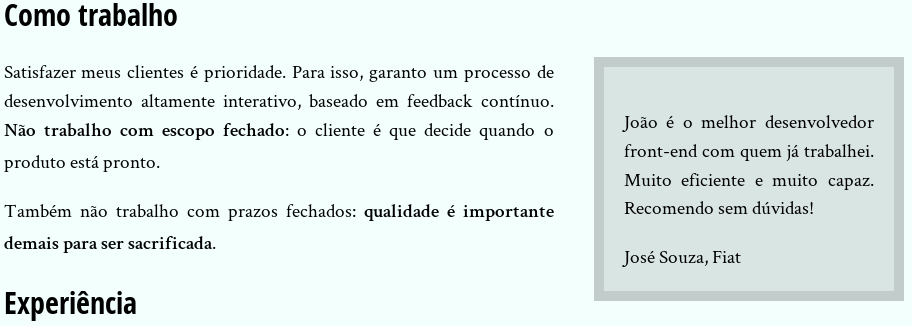
Para conseguirmos o resultado desejado, precisamos "quebrar" essa relação de ordem entre o HTML e o visual final, que podemos chamar de **fluxo do documento**.

Uma das propriedades que nos permite fazer essa "quebra" é a propriedade ***float***. Há dois valores possíveis para ela (além do none, que é o padrão): left e right. Quando colocamos float: left na citação, a caixa da citação permanece no mesmo lugar, mas o texto, que estava abaixo, passa a ficar *ao lado*da citação, como se tivéssemos definido tanto a citação como o texto como elementos *inline*.

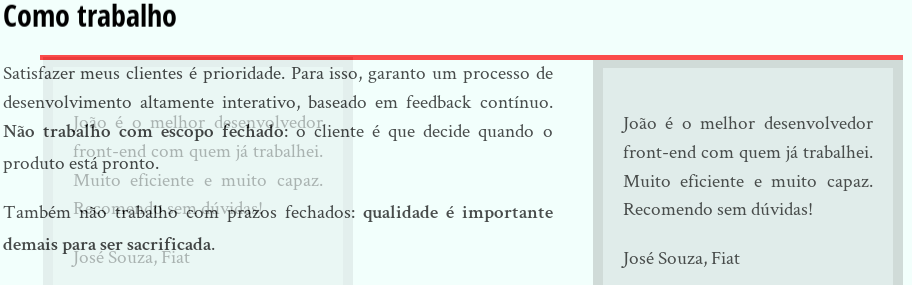


Mas repare que não é exatamente o mesmo comportamento de um elemento *inline*: o subtítulo seguinte, depois do parágrafo, também vai parar do lado da citação! O que aconteceu, na verdade, é que o resto do conteúdo da página "subiu", e a citação ficou **flutuando** do lado esquerdo do conteúdo.

Se usarmos o valor right, veremos que a citação **permanece na mesma altura do fluxo**, mas do lado direito do conteúdo.



É importante notar que os elementos flutuados permanecem na mesma altura de antes. Isso nos permite prever qual comportamento teremos ao usar a propriedade.



Para conseguir o resultado desejado no layout final, com cada citação de um lado, precisamos ser capazes de selecionar apenas uma delas individualmente. Para isso, podemos usar um *id* para cada uma delas mas, como vimos nas aulas anteriores, o ideal é **sempre usar classes**. Então podemos colocar mais uma classe nas citações:

<h2>Como trabalho</h2>

<blockquote class="citacao-bio-fiat">

<p>João é o melhor...</p>

<cite>José Souza, Fiat</cite>

</blockquote>

<!-- ... -->

<h2>Experiência</h2>

<blockquote class="citacao-bio-petrobras">

<p>João domina as tecnologias...</p>

<cite>Manoel Santos, Petrobrás</cite>

</blockquote>

Agora, nas classes recém-criadas, colocamos a declaração da propriedade float:

.citacao-bio-fiat {

float: right;

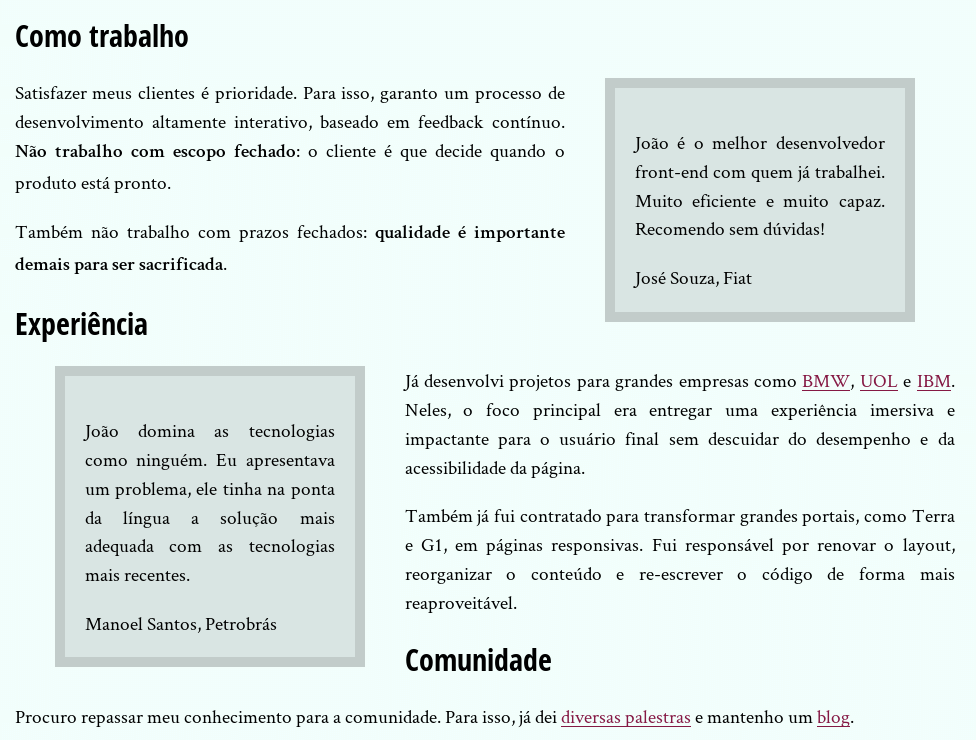
}

.citacao-bio-petrobras {

float: left;

}

E temos quase o resultado esperado.

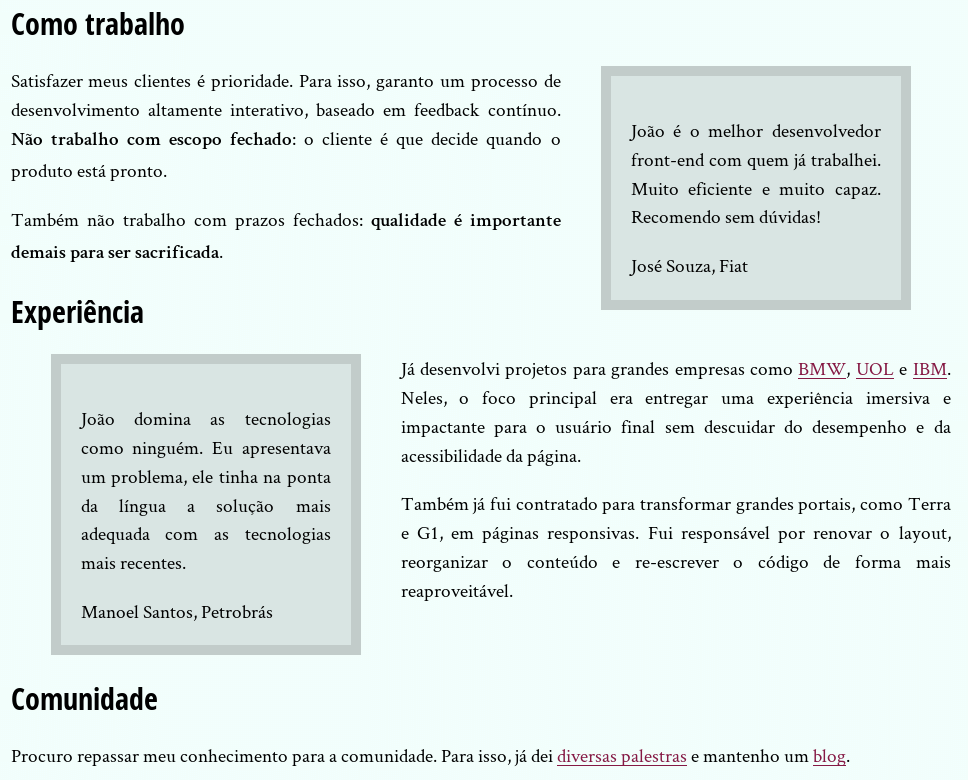


Repare que os subtítulos também acabam ficando do lado das citações; não queremos isso. Felizmente, podemos controlar o fluxo do documento com relação aos *floats* da página. Para isso, usamos a propriedade clear, que permite dizer que não queremos *floats* **do lado** de um determinado elemento. Por exemplo, se não queremos *floats* à esquerda dos nossos subtítulos, podemos fazer:

h2 {

clear: left;

}



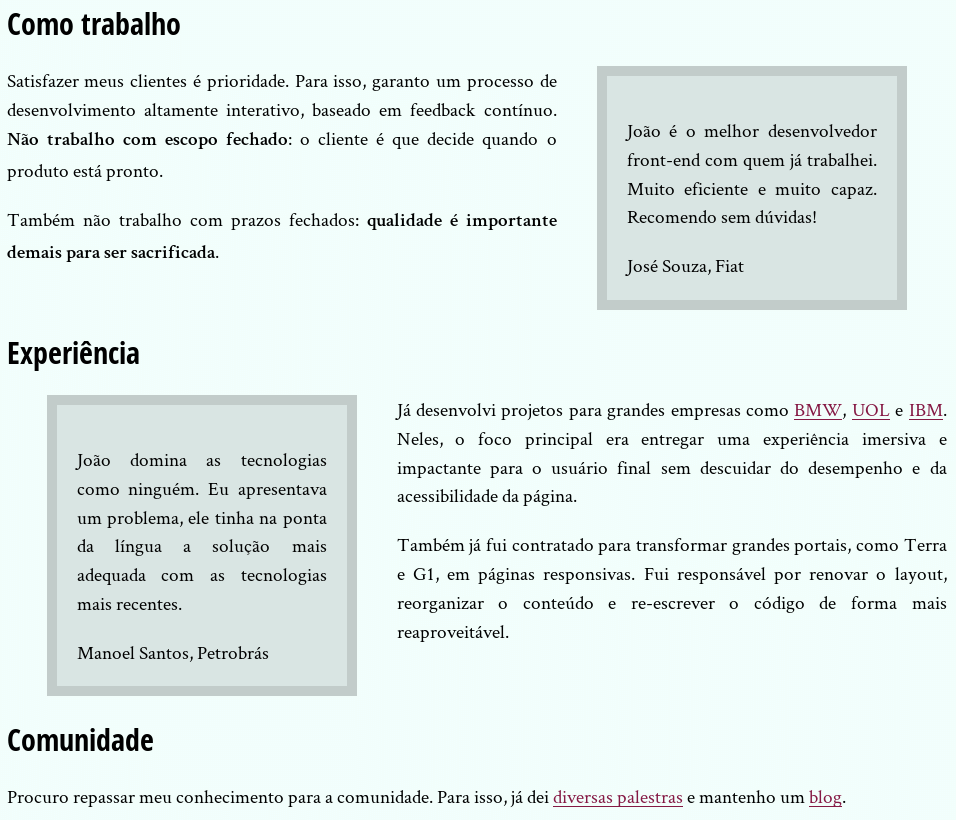
Repare que o subtítulo "Comunidade" é "empurrado" para baixo, para não ficar ao lado da citação com float: left. Mas o subtítulo "Experiência" continua ao lado da citação com float: right. Então, na verdade, queremos que não haja *floats* em nenhum dos lados dos <h2>. Para isso, usamos o valor both na propriedade clear:

h2 {

clear: both;

}

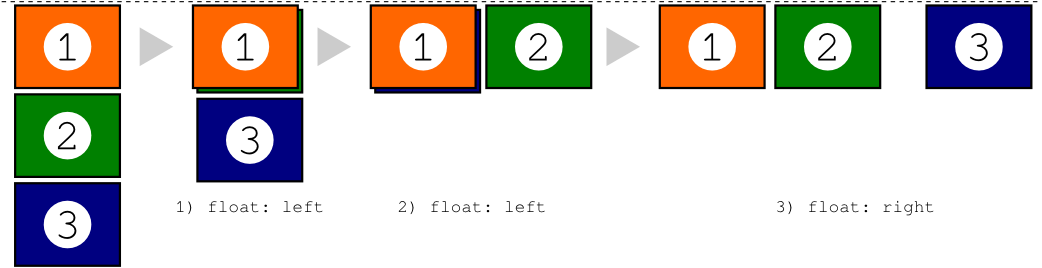
Assim, chegamos no resultado esperado, bastando apenas alguns pequenos ajustes nas margens das citações para que elas fiquem alinhadas com o texto.



## **Floats e mais floats**

Podemos aproveitar a ideia de *floats* para posicionar a foto e a barra lateral também. Para entender como isso é possível, precisamos entender o que acontece quando temos vários *floats* seguidos no HTML.

Como o *float* faz com que o elemento saia do fluxo, o resto do conteúdo da página sobe. Assim, se outro *float* vem logo em seguida no fluxo, esse outro *float*acabará **na mesma altura** do primeiro *float*, independente do lado em que ele vai ficar.



Agora, e se esse segundo *float* tem a propriedade clear definida justamente com o lado em que o primeiro *float* está?

.primeiro {

float: left;

}

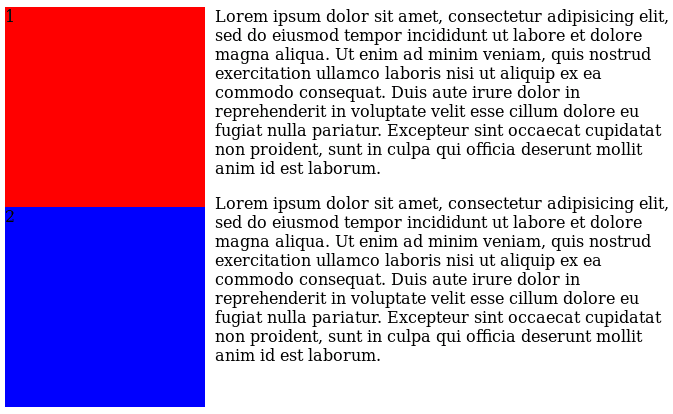
.segundo {

float: left;

clear: left;

}

Neste caso, o segundo *float* será "empurrado" para baixo, da mesma forma que qualquer outro conteúdo. Porém, o conteúdo sem *float* continuará ao lado **dos dois *floats***.



Se olharmos para o resultado do cenário descrito acima, veremos que ele é bem parecido com o resultado que desejamos, apenas espelhado. Então podemos tentar aplicar essa lógica no nosso layout, definindo também as larguras para o conteúdo, de forma que a parte do texto fique com 80% da largura da tela e a foto e a barra lateral com 20%.

main {

width: 80%;

}

img,

aside {

width: 20%;

float: right;

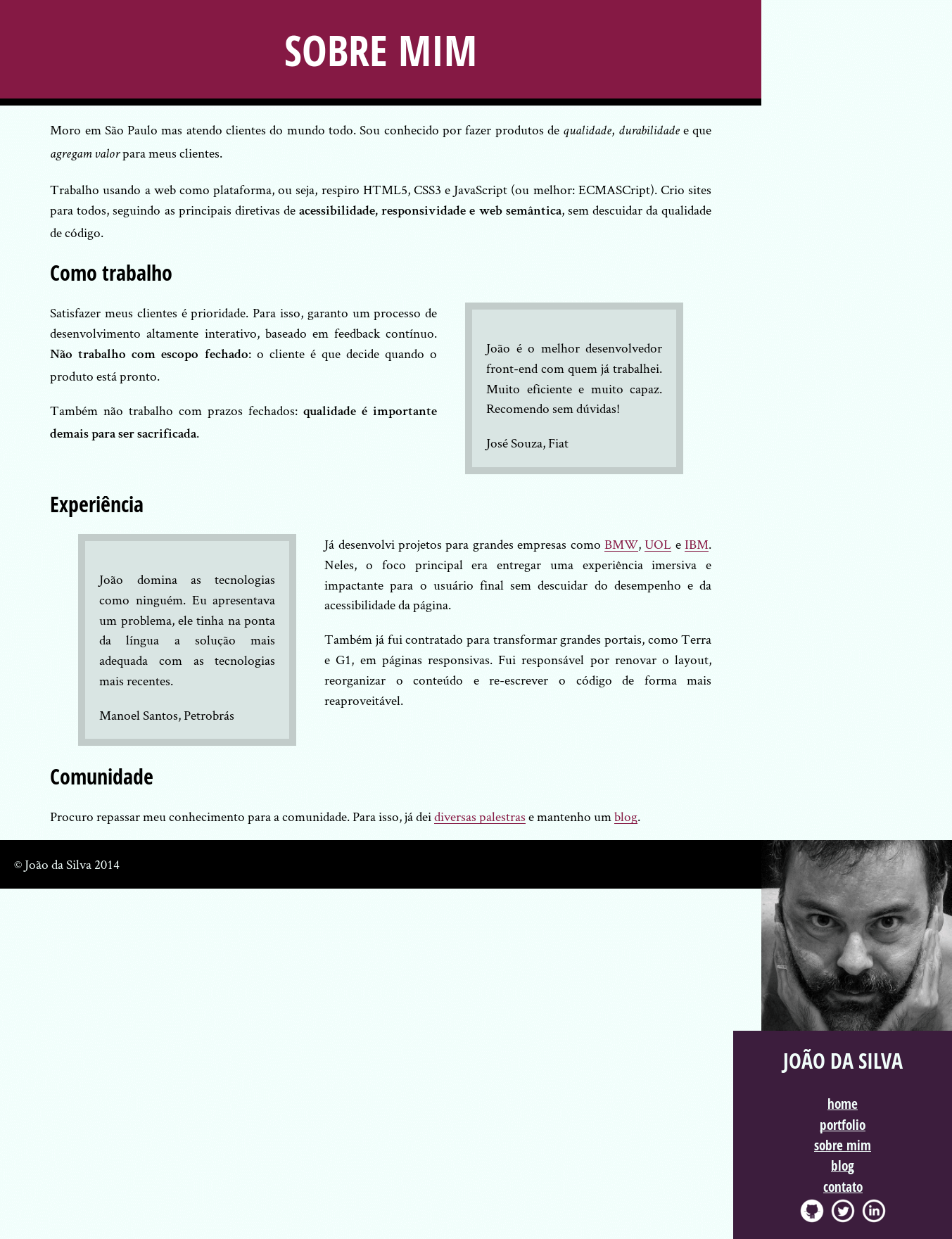
}

aside {

clear: right;

}

Porém, se fizermos isso, o resultado não será o esperado: a foto e a barra lateral ficam na mesma altura do rodapé. Além disso, a barra lateral fica mais larga que a foto.



Esse segundo problema ocorre devido ao *box model* padrão do navegador. Como a barra lateral tem *padding* e o *padding* não entra no cálculo da largura do elemento por padrão, ela acaba ficando mais larga. Para resolver esse problema, basta dizermos ao navegador que queremos que ele calcule a largura do elemento levando o *padding* em conta (e a borda, que colocaremos mais adiante) usando a propriedade box-sizing com o valor border-box:

aside {

box-sizing: border-box;

}

Isso resolve o problema da largura, mas não o fato de a imagem e a barra lateral estarem na altura do rodapé. Porém, na verdade, isso deveria acontecer, pois os elementos flutuados **permanecem na altura em que estavam** e a imagem, de fato, estava abaixo do texto, como está agora.

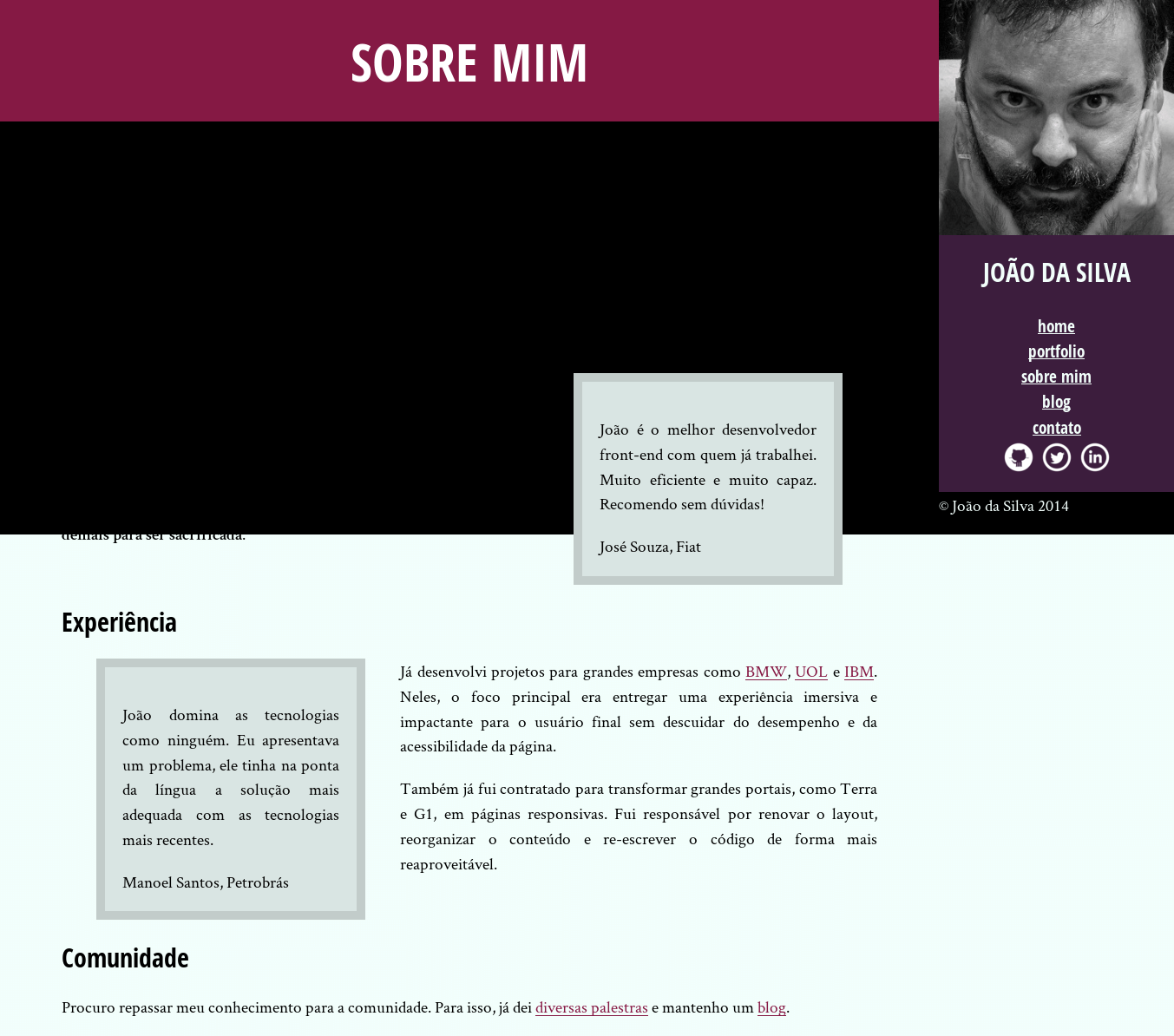
Como fazer, então, para que ela fique no topo da página? A solução é lembrar que ***floats* seguidos formam uma linha**. Se o <main> também for um *float*, a imagem ficará em linha com ele. Ou seja, basta adicionar a declaração:

main {

float: left;

}

Ao fazermos isso, a imagem e a barra lateral ficarão no lugar certo, ao lado do cabeçalho. Mas, estranhamente, o rodapé da página também sobe. Por que?



## **Cuidado com tantos floats!**

Precisamos lembrar que, sempre que transformamos um elemento em *float*, ele sai do fluxo da página, o que faz com que o resto do conteúdo *suba*. Neste caso, o rodapé subiu para ocupar o espaço entre os *floats* que criamos.

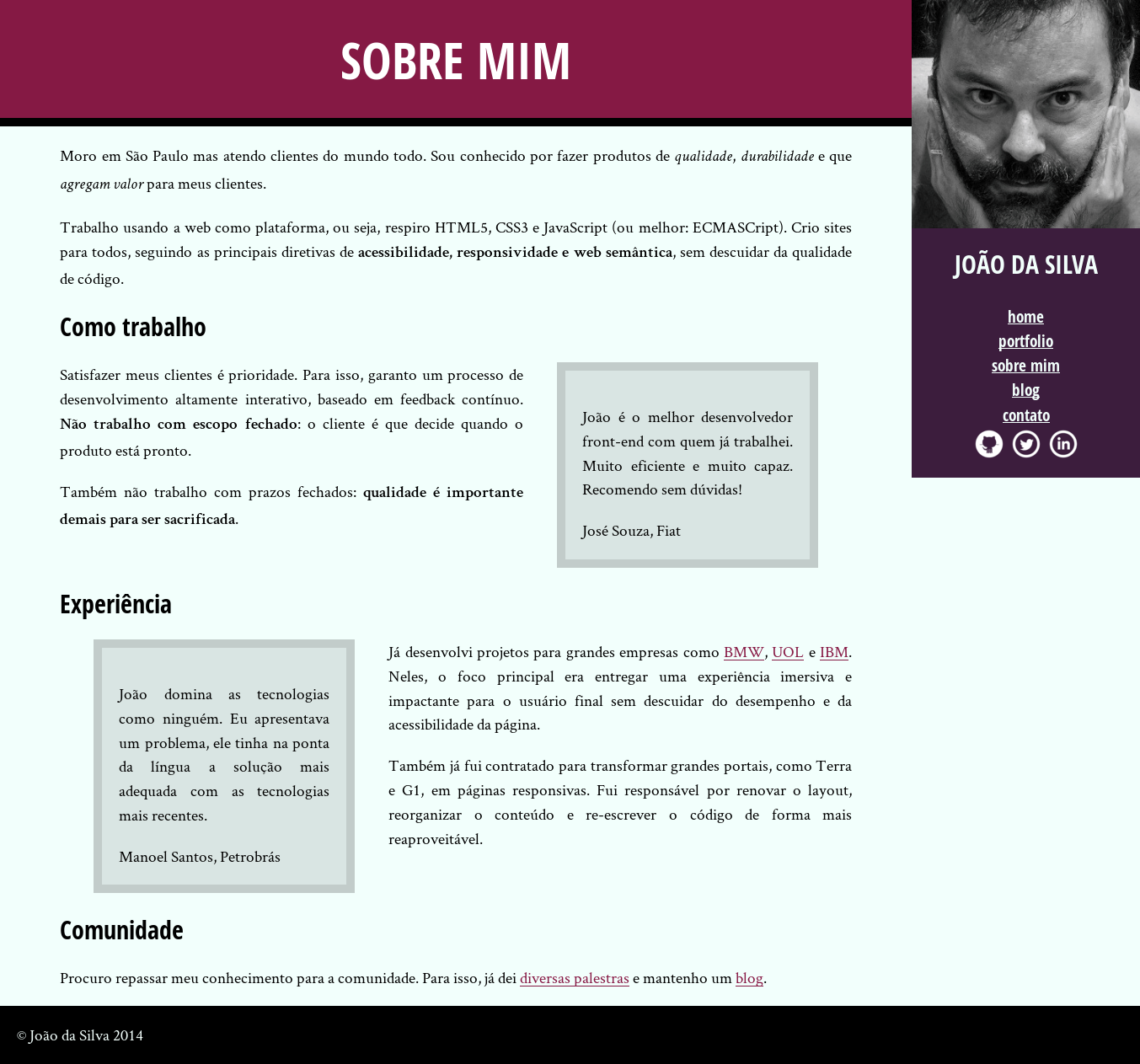
Para resolver esse problema, basta lembrarmos da propriedade clear. Não queremos nenhum *float* em nenhum dos lados do nosso rodapé, então podemos declarar:

footer {

clear: both;

}

Com isso, chegamos num resultado bem mais próximo do desejado.



Frequentemente, quando usamos *floats* para posicionar os elementos do nosso layout, precisamos "limpar" a sequência de *floats*, como no nosso caso agora. Assim, é comum encontrarmos declarações clear: both nesse tipo de layout. Essa é uma solução tão comum que até ganhou um nome popular: ***clearfix***. Frequentemente, projetos incluem em seus arquivos CSS uma classe cujo único propósito é fazer essa "limpeza" de floats:

.clearfix {

clear: both;

}

Se tivéssemos essa classe no nosso arquivo CSS, bastaria aplicá-la no nosso rodapé:

<footer class="clearfix">

&copy; João da Silva 2014

</footer>