Questões gerais:

- O que te excita ou te interessa em programação/codificação?

  O modo que a programação/codificação atua em nossas vidas no cotidiano, pois é incrivelmente grande. A tecnologia mais importante é aquela que nós não lembramos no dia-a-dia, pelo simples fato dela já fazer parte de nossa vida e de nós. Então seja em qualquer lugar, o meu interesse em programação está aí: codificação em todo lugar, tornando a nossa vida mais fácil e divertida.

- Fale sobre seu ambiente de desenvolvimento preferido. (SO, Editor, Browser, Ferramentas, etc)

  Um bom programador é aquele que independe de "muletas" para programar. É claro que existem preferências, e para mim não é diferente, mas o fato é que qualquer ferramenta que se torne uma muleta para o desenvolvedor é ruim. Entretanto, falando sobre um ambiente que me agrada: *costumo* desenvolver meus códigos front-end no **Sublime Text** – editor de texto – no sistema operacional Windows 7 – apesar de não ser o melhor – como de costume. Para pegar informações que só o protótipo pode dar – como cores hexadecimais, etc – utilizo o Adobe Fireworks CS5. Começo a desenvolver no Google Chrome, e a partir dele testo as páginas em outros navegadores em suas atuais e *antigas* versões – por exemplo, Opera, Safari, Mozilla Firefox e **Internet Explorer**. Gostaria de ressaltar que o que foi dito anteriormente são somente costumes, e que estou aberto para novas sugestões e ideias, das quais nos permitam aprender e desenvolver-nos cada vez mais e melhor.

- Descreva seu trabalho quando cria uma página web.

  Normalmente, antes de começar a desenvolver os códigos front-end, costumo analisar todo o protótipo procurando qual a parte ideal por onde começar – caso não haja uma sequência determinada para começar e/ou terminar. Daí em diante, começo a desenvolver página por página, me atentando o máximo aos detalhes do protótipo. Ao mesmo tempo em que desenvolvo o código do layout – incluindo responsividade – testo em outros navegadores, além de validar os códigos na W3C. Tudo terminado, mostro a uma pessoa que confere o projeto para aprovar ou fazer análises para possíveis mudanças e, após esse processo entrego o projeto.

- Descreva a diferença entre *progressive enhancement* and *graceful degradation*.

  Ambos visam fornecer conteúdo para qualquer tipo de tecnologia que está acessando seu site. A partir daí, a diferença é clara: *progressive enhancement* fornece conteúdo básico e funcionalidade de sua página web para todos os tipos de tecnologias que a acessam, e para quem acessa de uma tecnologia melhor/mais avançada, é concedida uma versão melhor da página. Agora, *graceful degradation* diz que a sua página ainda funcionará se for acessada por um usuário que utiliza uma tecnologia inferior.

- Explique o que significa "HTML Semântico".

HTML Semântico é um código HTML – ou sua estrutura – bem feito em um projeto, onde esse código pode, ou não, seguir algum padrão de desenvolvimento. Definir se o código está bem feito é definir se esse código utiliza tags HTML certas para fazer sentido por si só. Assim, é possível que mecanismos de busca entendam e filtrem melhor os conteúdos dessa página.

- Como você pode otimizar os recursos de um site?

  Podemos otimizar os recursos de um site de muitas formas, entre elas está a utilização de ferramentas que façam esse trabalho para você – mesmo existindo contraindicações. Outra forma é otimizar o site para pesquisa, o que chamamos de SEO. Neste, a utilização de, por exemplo, *metatags* se torna bastante importante. Podemos também otimizar recursos de um site para usuários mobile, fazendo-os enxergar todo o conteúdo do site de maneira que ele não force a visão para ter sucesso, tornando o site *responsivo*. Há também a possiblidade de escrevermos um código limpo e validado pela W3C, otimizando os recursos do site para ser melhor filtrado e achado nas buscas. Otimizar um site e seus recursos é muito mais do que descobrimos, é também se atentar aos mínimos detalhes que possam fazer o seu site se destacar ou sumir.

- Fale 3 formas de diminuir o *page load* (tempo de carregamento real e percebido)

  Escreva o seu código, copie-o em outro lugar e faça-o dele minificado. Comentários e outros são importantes no meio do código? Sim, mas não para o navegador. Então removendo esses espaços que não fazem diferença para o browser faz com que a página diminua seu tempo de carregamento.
  Evite utilizar links externos, como importação de arquivos JS de outro servidor pois o seu pode estar respondendo rápido, mas talvez o dos outros não.
  Não carregue todo o conteúdo da página de uma vez, e sim somente o começo, e somente se o usuário se interessar o suficiente para rolar a página mostre os seguintes conteúdos.

- Quais ferramentas você usa para testar a performance do seu código?

  Utilizo mais o Google Page Speed Insights, Load Impact e recentemente comecei a agregar o JMeter em fases, pois ainda estou o estudando.

- Se você pudesse dominar uma tecnologia, qual seria?

  Com certeza design responsivo, pois o mundo se não é já está se tornando e será mobile.

Questões específicas de HTML:

- O que um `doctype` faz?

  Fornece informações ao navegador sobre a página.

- Qual a diferença entre *standards mode* e *quirks mode*?

  Para a compilação de códigos de páginas, os navegadores utilizam dois modos, e erroneamente falando, são do modo antigo e correto/novo. O antigo – *quirks mode* – e o

novo/correto – *standards mode*. Enfim, o novo calcula o *Box Model* da maneira correta e o antigo não.

- Quais as limitações quando utilizamos páginas XHTML?

  Em páginas XHTML as regras são muito mais rígidas, como, por exemplo, ao não fechar uma tag, se não fechá-la, dá erro. Além disso, se a versão do XHTML não reconhecer novas tags, infelizmente não será possível a utilização dessas, como, por exemplo, a utilização de tags HTML5.

  - Existe algum problema em utilizar páginas como `application/xhtml+xml`?

    Somente se os códigos dessa página não *apoiarem* e não estarem seguindo os padrões da linguagem XHTML.

- Quais são os benefícios em utilizar o atributo `data-`?

  Dentre os benefícios está o de adicionar metadados em tags, permitindo adicionarmos atributos que nós mesmos poderíamos criar.

- Considere o HTML5 como uma plataforma web aberta. Quais são os blocos de construção de HTML5?

```
<!DOCTYPE html>
<html>
	<head>
		<meta charset="utf-8" />
		<title></title>
	</head>
	<body>

	</body>
</html>
```

- Descreva a diferença entre cookies, sessionStorage e localStorage.

  O *localStorage* armazena dados no computador do usuário. O *sessinStorage* faz a mesma coisa, entretanto somente durante a sessão, ou seja, quando ele fechar o navegador os dados armazenados serão perdidos. Os *cookies* são formas de fazer tudo o que foi dito acima, entretanto de uma maneira mais antiga.

Questões específicas de CSS

- Descreva o que é o arquivo "reset" do css e o que ele faz e como ele é útil.

  Um arquivo reset é um arquivo que limpa todos os valores pré-definidos de atributos. Então, quando vamos programar um CSS esses valores atrapalham, e com o reset é possível zerá-los e por fim especificarmos do nosso melhor modo.

- Descreva o que são floats e como eles funcionam.

  Float é uma propriedade CSS. Nela é possível atribuirmos os valores de *left, right, none* e *inherit*. Eles posicionam elementos da estrutura HTML em uma devida posição da tela. Por exemplo, se desejo posicionar uma div no lado esquerdo, atribuo o valor *left* para esse float; se desejar para o direito, atribuo o valor *right*; se não desejo atribuir nenhum valor de posicionamento, atribuo o *none* e por fim, atribuo o valor *inherit* caso seja necessário herdar essa propriedade de seu elemento pai – caso a div tenha.

- Quais são as várias técnicas para "clearing" e quais delas são apropriadas para qual contexto?

  As técnicas são: *none, left, right, both, initial* e *inherit*. Caso uma div que segundo o código estão dentro de outra div, mas na página não aparece dentro, necessitamos atribuir o valor *both* para arrumar o erro. Se desejarmos herdar a propriedade, atribuímos o valor *inherit*, se quisermos posicionar a div na esquerda ou na direita, atribuímos os valores *left* e *right*, respectivamente. Atribuímos *none* se quisermos que o elemento não flutue e, por fim, atribuímos o valor *initial* se desejarmos conceder ao elemento seus valores *default*.

- Explique o que são CSS Sprites, e como você implementaria eles em um website.

  CSS Sprites é uma técnica que combina várias imagens em uma só, visando melhorar a performance na hora de buscar uma imagem no servidor. Sim, implementaria em um website por esse mesmo motivo: a melhora de performance.

- Quais são suas técnicas favoritas para troca de imagens e quais dela você usa.

  Utilizo com mais frequência a técnica de *:hover*, entretanto, sempre busco aprender e a implementar novas técnicas.

- CSS Hacks, arquivo condicional .css, ou... alguma outra coisa?

  No desenvolvimento de um site é normal existir problemas, por exemplo, de compatibilidade, e a partir daí existem inúmeras soluções. Acredito que um bom desenvolvimento de site é aquele que de primeira consiga funcionar em todos – ou quase todos – navegadores. É necessário estudos e testes, mas é possível. Particularmente, quando não consigo resolver um problema que me obriga a escolher um tipo de solução, tento refazer a parte novamente, caso o problema ainda persista, dependendo do caso – se há inúmeras alterações por falta de compatibilidade – parto para a utilização de CSS hacks. Lembrando que sempre procuro a melhor solução para esses problemas e outros em um projeto.

- Como você desenvolve sua página para browsers com recursos limitados?

  Procuro atender aos requisitos de forma que se tornem leves para a página.

  - Quais técnicas/processos você usa?

Antes de tudo, procuro analisar todo o layout, e a partir daí consigo enxergar se necessitarei/poderei usar algum componente que seja distribuído ou se terei de desenvolver algo milimetricamente pensado, para não fazer com que a página se torne pesada ao ser carregada. Procuro também sempre *minificar* os arquivos CSS, JavaScript e HTML, pois espações em branco e comentários não são importantes para o browser, somente são para nós. Às vezes é *ruim* duplicarmos os mesmos arquivos, mas acredito que vale a pena.
Lembrando que sempre procuro a melhor solução para esses problemas e outros em um projeto.

- Quais são as diferentes formas de visualizar conteúdo escondido (e como fazer para deixar eles disponíveis apenas para leitores de tela?)

  É possível a utilização de *Media Queries*. Por exemplo: podemos colocar um conteúdo numa *div*, escondê-la (*display: none*) e caso a página for acessada por um tamanho de tela – *a especificar* – o conteúdo, a div, é mostrado (*display: block*).

- Você já utilizou algum sistema de grid, se sim, qual você prefere?

  Nunca implantei em um projeto, pois estou procurando dominar a técnica.

- Você já utilizou ou implementou media queries ou css's específicos para mobile?

  Sim.

- Qual sua familiaridade com SVG?

  Nunca o implantei, mas conheço um pouco da técnica e reconheço que nos ajuda na melhora de performance.

- Como você optimiza suas páginas para impressão?

  Media Queries: PRINT.

- Quais são algumas técnicas para escrever um eficiente CSS?

  Podemos falar de pré-compiladores de CSS – aqueles que te ajudam a escrever um código mais rápido e eficiente – e utilizar padrões pré-definidos para/com a equipe de desenvolvimento.

- Você já utilizou pré-processadores css? (SASS, Compass, Stylus, LESS)

  Somente SASS.

  - Se sim, descreva o que você gostou e o que não gostou com eles.

    Não utilizei o máximo que o SASS nos concede, mas o que gostei bastante é o método de atualização do arquivo *.scss* para o *.css*, além da declaração de variáveis que nos auxiliam

durante todo o projeto. Também gostei dos possíveis erros que ele nos informa caso um código possa dar errado, e quando realmente dá.

- Como você implementaria um website que não utilizaria fontes padrões nos computadores?

  Colocaria as fontes dentro de uma pasta do projeto e as importaria em meu arquivo CSS.

- Explique como um browser determina quais os elementos que correspondem a um seletor CSS.

  O browser possui seus elementos, e esses elementos estão especificados no arquivo CSS, então. É possível selecionar esse elemento pelo nome de sua tag – *body*, por exemplo – ou definindo se o elemento recebe um *id* e/ou *class*. A partir daí, é só codificar os estilos.

Questões específicas de jQuery:

- Explique o que é "chaining".

  É a técnica de aplicar mais de um método e/ou ação sobre um mesmo elemento.

- Explique o que é "deferreds".

  É uma técnica que consiste em passar um método que seja opcional.

- Quais são algumas especificações de optimização do jQuery que você pode implementar?

  Armazenar o objeto jQuery em uma variável; evitar a utilização do DOM; ser o mais possível específico em um seletor; encadear métodos; evitar repetir códigos; entre outros.

- O que o `.end()` faz?

  Ele pega a operação de filtragem mais recente e retorna os elementos ao seu estado anterior.

- Como, e porque, faria namespacing de vários agregadores de eventos?

  Faria da seguinte forma:

  ```
  $(“#iddoelemento”)
      .on(“evento”, ação);
  ```

- Cite 4 valores diferentes que você pode passar pelo método jQuery

  Seletores, HTML, array, objeto e outros.

- Quais são os efeitos do queue?

  Queue é uma fila, onde podemos armazenar, por exemplo, objetos. É necessário saber que a queue funciona da seguinte forma: o primeiro objeto – nesse caso – que entra é o primeiro objeto a ser retirado.

- Qual a diferença entre `.get()`, `[]`, e `.eq()`?

  O *.eq()* retorna um objeto jQuery, o *.get()* retorna um elemento – DOM – e *[]* você pega o atual valor.

- Qual a diferença entre `.bind()`, `.live()`, e `.delegate()`?

  A diferença é que o *bind()* registra os eventos apenas nos elementos que estão no DOM, o *.live()* permite que os eventos sejam registrados para elementos que virão via *ajax* e/ou *javascript* e o *delegate()* é similar, entretanto não utiliza o *documento* como elemento raíz, como o *live()* utiliza.

- Qual a diferença entre `$` e `$.fn`? Ou, apenas, o que é `$.fn`.

  $.fn é utilizado para a criação de plug-ins. A diferença entre *$* e *$.fn* é que na segunda opção quando você atribuir uma função para o seu plugin essa função estará disponível para todas as instâncias do objeto.

- Optimize esse seletor:

```
$(".foo div#bar:eq(0)")

Otimização:

$("#bar.foo")

   $("eq(0)");
```

- Qual a diferença entre 'delegate()' e 'live()'?

  O *.live()* permite que eventos sejam registrados para elementos que virão via *ajax* e/ou *javascript* e o *delegate()* é similar, entretanto não utiliza o *documento* como elemento raíz.