Disciplina: Aplicações Ricas

Diretoria Acadêmica
Centro Universitário Senac

Prof. **Mario L. P. Toledo**

Correção dos Exercícios

- arquivos HTML não podem ter elementos com o mesmo id
- caminho para CSS tem que estar a partir do index (um dos projetos tinha apontando para uma pasta, mas o arquivo estava na raiz)
- títulos tem que estar dentro da section
- tags escritas errados (uma delas estava escrito foolter)
- tags de footer, header, nav, tem que estar dentro da body sempre
- se forem compactar o envio, me enviar o arquivo sempre em .zip apenas
- prestar atenção no doctype, tem que ser apenas <!DOCTYPE html>

Não esqueçam sempre de revisar o código e testar no navegador antes de enviar.

Caso alguém queira corrigir a atividade e me reenviar, eu deixei aberto a resubmissão pelo blackboard.

Uma sugestão é testar o seu código pelo validador do w3c: https://validator.w3.org/

Aula 2 - Formulários e Transições + Animações

Formulários

- Campos no HTML4 são limitados
  - Apesar de haverem diferentes tipos de inputs, campos de texto, normalmente, são type="text"
  - Campos como telefone, data, hora, busca e etc são representados como campo padrão de texto
- Novos campos de entrada no HTML5 trazem comportamentos e formatação de dados mais úteis além de texto puro

Formulários - Números

- E se quisermos um campo só para representar números?
- Em HTML5, existe o tipo de input number
  - <input type="number" value="4">
  - <input type="number" value="4" step="2">
- Resultado visual no Chrome:

Formulários - Números

- Para valores que não exijam precisão, é possível utilizar o tipo de input range
  - <input type="range" min="1" max="10" value="2">
  - <input type="range" min="0" max="1000" value="20" step="20">
- Resultado visual no Chrome:

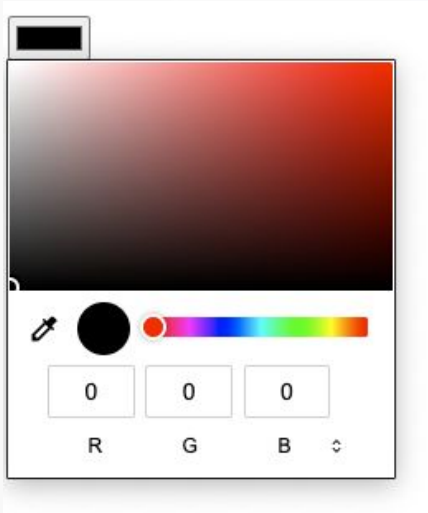
Formulários - Data e Hora

- Com o HTML5, temos várias maneiras de solicitar input do usuário para datas ou horas

- Horas: <input type="time" value="10:40">
- Data: <input type="datetime" value="2022-03-28">
- Data: <input type="datetime" value="2022-03-28">
- Data e Hora: <input type="datetime-local" value="2010-05-31T21:00"> (datetime foi removido)
- Mês: <input type="month" value="2010-05">
- Semana: <input type="week" value="2014-W20">

Formulários - Color Picker

- Para seleção de uma cor, usar o type="color"
  - <input type="color" >
- Resultado visual no Chrome:



Formulários - Combo Box

- Parecido com o select, mas também permite que o usuário digite o valor
- Elemento <datalist>
  - Utilizado em conjunto com o <input> de texto
  - São adicionadas opções como <option>

```
<input type="text" name="browser" list="browsers">
<datalist id="browsers">
    <option value="Internet Explorer">
    <option value="Firefox">
    <option value="Safari">
    <option value="Chrome">
    <option value="Opera">
</datalist>
```

Formulários - Textos

- Outros campos de texto para contextos semânticos e comportamentos em outros dispositivos
  - tel: Número de telefone
    - Agentes podem fazer integração com agenda de contato
  - search: Campo de busca
    - Agentes podem mudar aparência e comportamento
  - email: Campo de Email
    - Pode ter formatação e validação. Agentes podem promover integração com agenda de contato
  - url: Campo de Url
    - Pode ter validação e formatação

Formulários - Outros Atributos

- autofocus: O foco será colocado neste campo automaticamente ao carregar a página.
- placeholder: Adiciona um texto introdutório no campo de entrada
- autocomplete: Ativa o autocomplete de um campo de texto

Formulários - Validação

- Validação no HTML4 necessitava ser feito manualmente
  - Necessidade de código JavaScript para fazer validação
- Validação só acontece ao enviar dados por formulários
- Pode ser ativada ou desativada
- Permite validação por JavaScript

Formulários - Validação

- required: torna um campo de formulário obrigatório (valor precisa ser preenchido)

```
<input name=”login” required>
```

- maxlength: define um valor máximo de caracteres para um campo de texto

```
<input name=”name” maxlength="50">
```

- minlength: define um valor mínimo de caracteres para um campo de texto

```
<input name=”name” minlength="3">
```

Formulários - Validação

- pattern: permite definir expressões regulares de validação, sem Javascript

```
<input name=”CEP” id=”CEP” required pattern=”\d{5}-?\d{3}” />
```

- novalidate: atributo para inserir no <form> e definir que ele não será validado ao enviar o formulário
- formnovalidate: atributo para inserir em um <button> ou <input> de submit, indicando que não é para validar um formulário em seu envio

Formulários - CSS de Validação

```
input:invalid {
    border: 1px solid red
}
input:valid {
    border: 1px solid black
}
```

Outras opções incluem required e optional

Formulários - Validação Customizada

- Permite criar a validação customizada de um campo de formulário
- Pode ser usada em conjunto com o evento oninput
- É definido pelo método setCustomValidity, que precisa ser chamado manualmente pelo JavaScript.

```
<form>
    <input id="campo1" type="text" required>
    <input type="submit">
</form>
<script>
    var fldName = document.getElementById('campo1');
    fldName.oninvalid = function () {
        fldName.setCustomValidity("");
        if (!fldName.validity.valid) {
            fldName.setCustomValidity("Campo obrigatório");
        }
    };
</script>
```

Formulários - Validação Customizada

- Inputs também podem ser chamados manualmente para validação através do método checkValidity

```
<input id="email" type="email">
<input type="button"
onclick="document.getElementById('email').checkValidity();"
value="Validar">
```

Outras validações:
https://html.spec.whatwg.org/multipage/form-control-infrastructure.html#the-constraint-validation-api

Formulários - Saída de Dados

- É possível exibir um feedback para o usuário enquanto ele preenche o formulário
- Uma das formas é utilizar o elemento <output>
  - Para utilizá-lo, é necessário especificar quais inputs serão utilizados para exibir o retorno através do atributo for
  - Por fim, é necessário utilizar Javascript no elemento pai (o próprio <form> ou um <fieldset>)

```
<form oninput="o.value = a.valueAsNumber + b.valueAsNumber">
    <input name="a" id="a" type="number"> +
    <input name="b" id="b" type="number"> =
    <output name="o" for="a b"></output>
</form>
```

Formulários - Saída de Dados

- Nas aulas anteriores vimos três elementos:
  - <progress>
  - <meter>
  - <details>
- Por que são importantes?
  - Estes elementos também podem ser utilizados como saída de dados para fornecer informações importantes para o usuário

Formulários - Saída de Dados

- O elemento <progress> pode ser utilizado para indicar o progresso do usuário no preenchimento de uma sequência de formulários
  - <progress value="5" max="9">Página 5 de 9</progress>
- Resultado visual no Chrome:

Formulários - Saída de Dados

- O elemento <meter> é parecido com o <progress>, mas é de uso mais geral
  - Exemplos?
- Devemos especificar o mínimo e máximo (min e max)
- Possui também os atributos low, high, e optimum
  - <meter value="3" min="0" max="7">3 de 7</meter>
- Resultado visual no Chrome:

Formulários - Saída de Dados

- O elemento de <details>, como vimos, contém informações adicionais que ficam ocultas em uma caixa até o usuário clicar na seta
- Pode estar dentro de formulários, contendo campos não obrigatórios
- É possível utilizar o elemento <summary> para esclarecer o que o elemento contém
  - Deve ser o primeiro elemento dentro do <details>

```
<details>
    <summary>Arquivo opcional:</summary>
    <input type="file">
</details>
```

Formulários - Saída de Dados

- Criar um formulário de pesquisa de satisfação em 3 partes:
  - A primeira parte deve solicitar informações pessoais (Ex.: nome, email, data de nascimento, página pessoal, número de telefone, endereço, etc.)
  - A segunda parte deve ter perguntas sobre preferências de produtos e serviços (Ex.: navegador preferido, cor preferida, etc.)
    - Utilizar <select>s quando o usuário for obrigado a selecionar um valor em uma lista pré-determinada
    - Utilizar <datalist>s quando o usuário puder selecionar um valor diferente das opções dadas
  - A terceira parte deve conter campos para texto livre (Ex.: comentários, sugestões, etc.)
- Utilizar o máximo possível de elementos vistos em aula

Transições

- CSS3 permite adicionar animações como transições em páginas HTML
- Animações acontecem quando há mudanças nas propriedades CSS
- Permite alterar propriedades CSS como uma transição através do tempo
- Propriedade transition

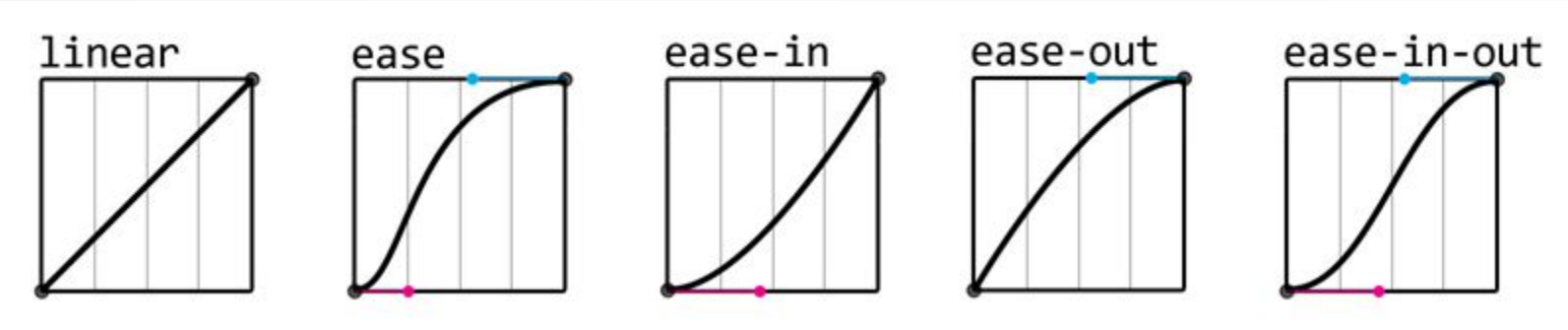
```
transition: width 2s;

transition: width 2s, height 4s;
```

Transições

- Pode também ser definido entre cada propriedade

```
div {
  transition-property: width;
  transition-duration: 2s;
  transition-timing-function: linear;
  transition-delay: 1s;
}
```



Animações

- CSS3 permite adicionar animações customizadas em elementos
- Faz a troca gradual de um estilo para outro
- É necessário especificar a animação de um quadro para o outro usando a propriedade @keyframes

```
@keyframes example {
  from {background-color: red;}
  to {background-color: yellow;}
}

div {
  width: 100px;
  height: 100px;
  background-color: red;
  animation-name: example;
  animation-duration: 4s;
}
```

Animações

- Também é possível alterar diferentes propriedades
- Animação também pode ser definida relativa ao estado atual da animação

```
@keyframes example {
  0%   {background-color:red; left:0px; top:0px;}
  25%  {background-color:yellow; left:200px; top:0px;}
  50%  {background-color:blue; left:200px; top:200px;}
  75%  {background-color:green; left:0px; top:200px;}
  100% {background-color:red; left:0px; top:0px;}
}

div {
  width: 100px;
  height: 100px;
  position: relative;
  background-color: red;
  animation-name: example;
  animation-duration: 4s;
}
```

Animações

- animation-name: nome da animação definido no keyframes
- animation-duration: duração da animação
- animation-delay: tempo de atraso para início da animação
- animation-iteration-count: define o número de vezes que uma animação é executada antes de parar
- animation-direction: define se a animação é normal, forward, backward ou alternate
- animation-timing-function: define a função de progresso da animação em relação ao tempo
- animation-fill-mode: define como uma animação aplica o estilo ao elemento antes e depois da execução da animação (norma, forward, backward ou both)

- animation: forma rapida de definir uma animação com todos os parâmetros:

```
animation: example 5s linear 2s infinite alternate;
```