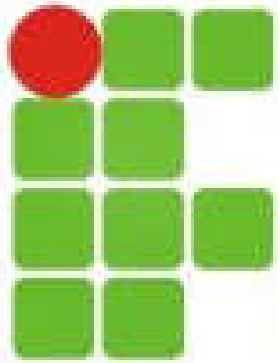
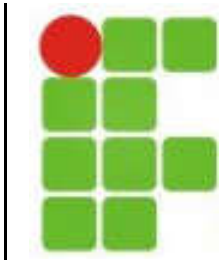
Fundamentos de Web Design 2

Professor Eng. Dr. Will Roger Pereira

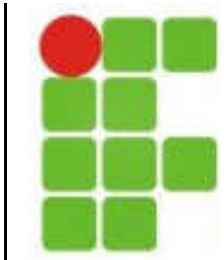


Objetivos desta aula



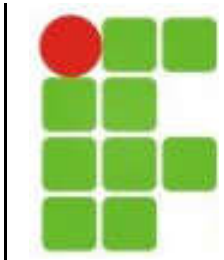
- Manipular texto de um elemento HTML;
- Realizar uma ação para cada elemento encontrado pelo seletor;
- Manipular código HTML de um elemento HTML;
- Manipular valores;
- Manipular atributos;
- Adicionar novo conteúdo HTML;
- Remover conteúdo HTML;
- Manipular CSS;
- Manipular classes CSS de elementos;
- Manipular dados nos elementos HTML – Atributos Customizados.

Introdução



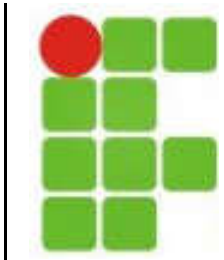
- Com jQuery, é possível manipular DOM de maneira muito mais fácil do que com Javascript.
- Afinal, não foi para isto que jQuery foi criado?
- Como sempre, os seletores serão muito importantes.

Manipular texto de um elemento HTML



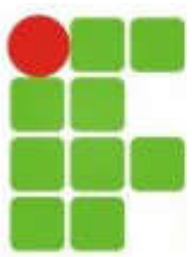
- Esta funcionalidade irá permitir a recuperação(get) e mudança(set) de **texto** de um elemento HTML;
- Qualquer tag HTML será ignorada;
- Para realizar isto, é necessário utilizar a ação **text** em um seletor;
- RECOMENDAÇÃO DE USO: Elementos que não possuem outras tags.

Recuperar texto de um elemento HTML



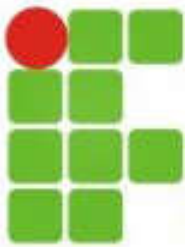
- Lembre-se: As tags HTML existentes serão ignoradas;
- Se o seletor encontrar diversos elementos HTML, o texto de todas será concatenado e retornado;
- Para contornar isto, olhe a seção → Realizar uma ação para cada elemento encontrado pelo seletor;
- Sintaxe: SELETOR.**text**();

Modificar texto de um elemento HTML



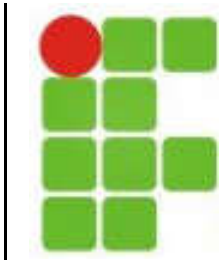
- As tags HTML contidas na string serão mostradas como texto;
- Se o seletor encontrar diversos elementos HTML, todos serão alterados;
- Sintaxe: SELETOR.**text**(string);
- O argumento da ação **text** representará o novo texto dos elementos.

Realizar uma ação para cada elemento encontrado



- Esta funcionalidade irá permitir que uma ação seja realizada para cada elemento encontrado pelo seletor;
- O argumento da ação **each** é uma função com um argumento: O índice do elemento;
- Se o seletor encontrar diversos elementos HTML, o índice 'i' será único para cada um;
- Para referenciar o elemento que está sendo processado, utilize o seletor $(this) dentro da função.
- Sintaxe: SELETOR.**each**(function(i){ ... });

Manipular código HTML de um elemento HTML



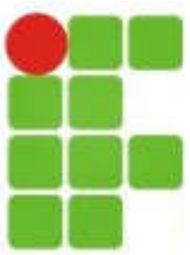
- Esta funcionalidade irá permitir a recuperação(get) e mudança(set) de código HTML de um elemento HTML.

- Ao contrário da manipulação de texto, qualquer tag HTML será considerada;

- Para realizar isto, é necessário utilizar a ação **html** em um seletor;

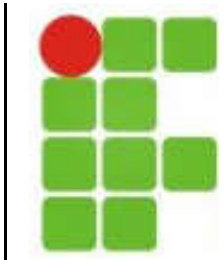
- RECOMENDAÇÃO DE USO: Elementos que possuem outras tags.

Recuperar código HTML de um elemento HTML



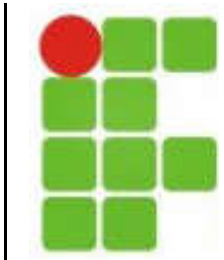
- As tags HTML existentes serão consideradas;
- Se o seletor encontrar diversos elementos HTML, somente o código do primeiro elemento encontrado será retornado;
- Utilize o **each** para retornar o de todos;
- Sintaxe: SELETOR.**html**();

Modificar código HTML de um elemento HTML



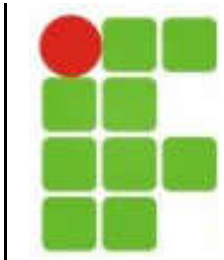
- As tags HTML contidas na string serão incorporadas ao código HTML do elemento;
- Se o seletor encontrar diversos elementos HTML, todos serão alterados;
- Sintaxe: SELETOR.**html**(string);
- O argumento da ação **html** representará o novo código HTML dos elementos.

Manipular valor de um elemento HTML



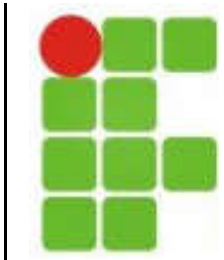
- Esta funcionalidade irá permitir a recuperação(get) e mudança(set) do valor de um elemento HTML;
- Para realizar isto, é necessário utilizar a ação **val** em um seletor;
- O valor está associado a elementos de entrada, como input dos tipos text, checkbox, radio, password, além do tipo select, dentre outros;
- RECOMENDAÇÃO DE USO: Alterar a entrada de elementos de entrada HTML.

Recuperar valor de um elemento HTML



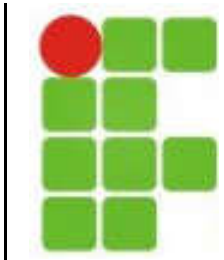
- É necessário utilizar a ação sem nenhum argumento, ou seja, com aridade 0;
- Se o seletor encontrar diversos elementos de entrada HTML, somente o valor do primeiro elemento encontrado será retornado;
- Utilize o **each** para retornar o de todos.
- Sintaxe: SELETOR.**val**();
- O que acontece se o valor for retornado de um elemento que não possui tal propriedade?

Modificar valor de um elemento HTML



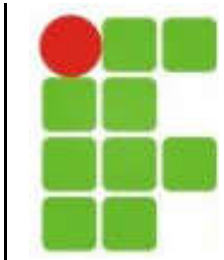
- É necessário alimentar a ação com o novo valor que se deseja para o elemento de entrada;
- Se o seletor encontrar diversos elementos HTML, todos serão alterados;
- Sintaxe: SELETOR.**val**(novovalor);
- Obs: Isto irá mudar o valor existente no elemento de entrada:
  - Mudará o conteúdo do elemento de texto;
  - Mudará a opção selecionada.
- O que acontece se o valor atribuído não existir como opção?

Manipular atributo de um elemento HTML



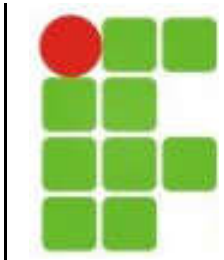
- Esta funcionalidade irá permitir a recuperação(get) e mudança(set) de um atributo de um elemento HTML;
- Para realizar isto, é necessário utilizar a ação **attr** em um seletor;
- O atributo está associado a características de elementos, como destino de um link, fonte de uma imagem, etc;
- RECOMENDAÇÃO DE USO: Alterar imagens, destino de links, ids... ou seja, qualquer atributo HTML.

Recuperar atributo de um elemento HTML



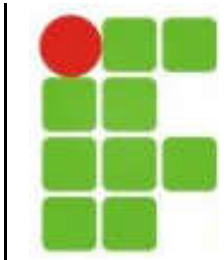
- É necessário alimentar a ação com o nome do atributo que se deseja recuperar o valor;
- Se o seletor encontrar diversos elementos HTML, somente o valor do atributo do primeiro elemento encontrado será retornado;
- Utilize o **each** para retornar o de todos.
- Sintaxe: SELETOR.**attr**(atributo);
- OBS: Se o valor retornado for undefined, o elemento não possui tal atributo.

Modificar atributo de um elemento HTML



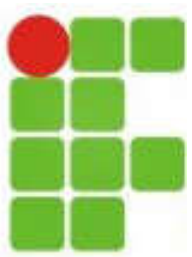
- É necessário alimentar a ação com o nome do atributo que se deseja recuperar o valor e o novo valor para este atributo;
- Se o seletor encontrar diversos elementos HTML, todos serão alterados;
- Para um atributo: SELETOR.**attr**(atributo,novovalor);
- Para muitos atributos: SELETOR.**attr**({atributo1:novovalor, atributo2:novovalor, ..., atributoN:novovalor});

Manipular conteúdo HTML



- Esta funcionalidade irá permitir a adição e remoção de conteúdo HTML de forma dinâmica em sua aplicação;
- Assim sendo, pode-se compor a página HTML conforme a aplicação vai sendo executada e os eventos vão disparando;
- A mudança só acontece no âmbito do navegador. Nenhuma mudança refletirá no código;
- Basta uma simples atualização para que a página volte ao normal.

Adicionando conteúdo HTML

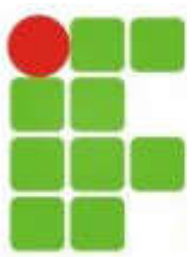


- É possível adicionar conteúdo de 4 maneiras diferentes:
  - Ao final de um seletor, como um conteúdo filho → **append**;
  - Ao começo de um seletor, como um conteúdo filho → **prepend**;
  - Depois de um seletor → **after**;
  - Antes de um seletor → **before**.

- OBS: Em qualquer uma das situações, se for encontrado mais de um seletor, os elementos são adicionados em todos;

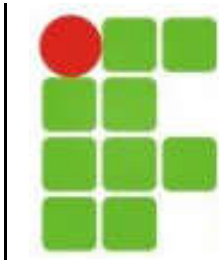
- OBS: Para adicionar diversos conteúdos de uma vez, separe os conteúdos por vírgula. Ex: SELETOR**.append**(conteúdo1, conteúdo2, ..., conteúdoN);

Adicionando ao final como conteúdo filho



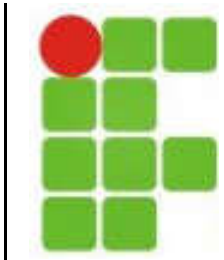
- Permitirá adicionar conteúdo, tratado como filho, no final do seletor;
- Ótimo para adicionar linhas a uma tabela, elementos a uma lista, parágrafos a uma div, imagens a uma galeria, etc;
- Lembrando, esta maneira adicionará ao final, como último filho do(s) elemento(s) selecionado(s);
- Sintaxe: SELETOR.**append**(conteúdo);

Adicionando ao começo como conteúdo filho



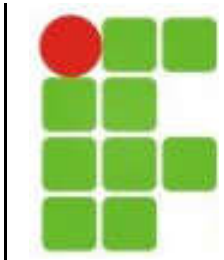
- Permitirá adicionar conteúdo, tratado como filho, no começo do seletor;
- Ótimo para adicionar linhas a uma tabela, elementos a uma lista, parágrafos a uma div, imagens a uma galeria, etc;
- Lembrando, esta maneira adicionará ao final, como primeiro filho do(s) elemento(s) selecionado(s);
- Sintaxe: SELETOR.**prepend**(conteúdo);

Adicionando depois de um seletor



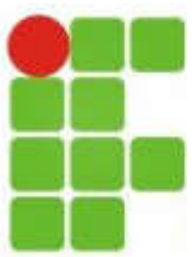
- Permitirá adicionar conteúdo, tratado como irmão, depois do seletor;
- Ótimo para adicionar elementos depois de um determinado elemento, em listas, tabelas, etc;
- Lembrando, esta maneira adicionará imediatamente depois do seletor;
- Sintaxe: SELETOR.**after**(conteúdo);

Adicionando antes de um seletor



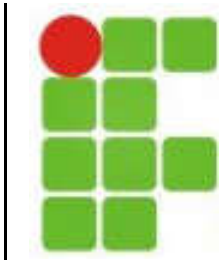
- Permitirá adicionar conteúdo, tratado como irmão, antes do seletor;
- Ótimo para adicionar elementos antes de um determinado elemento, em listas, tabelas, etc;
- Lembrando, esta maneira adicionará imediatamente antes do seletor;
- Sintaxe: SELETOR.**before**(conteúdo);

Removendo conteúdo HTML



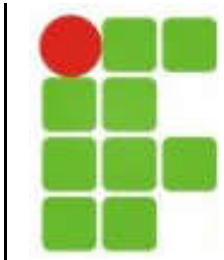
- É possível remover elementos de 3 maneiras diferentes:
  - Remover todos os elementos e seus filhos;
  - Remover todos os filhos;
  - Remover todos os filhos que se encaixarem em um seletor.

Removendo todos os elementos e seus filhos



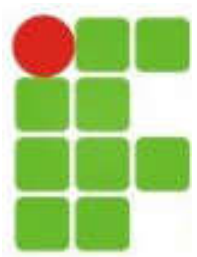
- Todos os elementos que se encaixarem no seletor serão removidos, juntamente com seus filhos;
- Tome cuidado, pois irá eliminar tudo que for contemplado pelo seletor utilizado;
- Ótimo para ser utilizado com os subseletores, por serem mais específicos;
- Sintaxe: SELETOR.**remove**().

Removendo todos os filhos



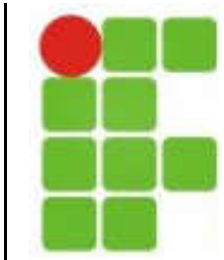
- Todos os filhos, sem exceção, dos elementos que se encaixarem no seletor serão removidos;
- Os elementos não serão removidos, somente seus filhos;
- É como se eles fossem esvaziados!!!
- Sintaxe: SELETOR.**empty**();

Removendo elementos com filtro



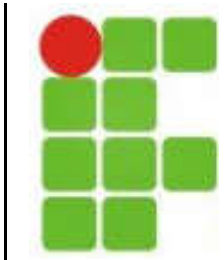
- Removerá os elementos que se encaixarem no **seletor**;
- O **seletor** será o argumento da ação remove neste caso. Será denominado ‘filtro’;
- Somente os elementos que forem contemplados pelo ‘filtro’ serão removidos. O restante será preservado;
- Sintaxe: SELETOR.**remove**(filtro).

Manipular CSS de um elemento HTML



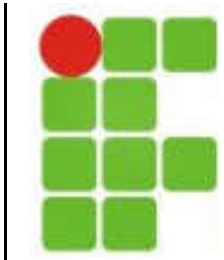
- Esta funcionalidade irá permitir recuperar e alterar valores de todos os atributos CSS, como cor, cor de fundo, etc;
- Para isto, é necessário utilizar a ação **css** em um seletor;
- Qualquer atributo CSS poderá ser manipulado através disto;
- O que acontecerá é que o CSS será alterado na memória do computador, e não afetará nada no código fonte da página;
- RECOMENDAÇÃO DE USO: Modificar o estilo dos elementos HTML de maneira dinâmica.

Recuperar CSS de um elemento HTML



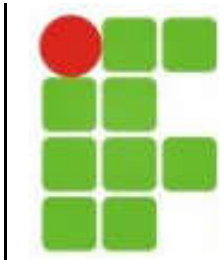
- É necessário alimentar a ação com o nome do atributo CSS que se deseja recuperar o valor;
- Se o seletor encontrar diversos elementos HTML, somente o valor do atributo CSS do primeiro elemento encontrado será retornado;
- Utilize o each para retornar o de todos;
- Sintaxe: SELETOR.**css**(atributoCSS).

Modificar CSS de um elemento HTML



- É necessário alimentar a ação com o nome do atributo que se deseja modificar o valor e o novo valor para o atributo CSS;
- Se o seletor encontrar diversos elementos HTML, todos serão alterados;
- Para um atributo: SELETOR.**css**(atributoCSS,novovalor);
- Para muitos: SELETOR.css({atributoCSS1:novovalor, atributoCSS2:novovalor, ..., atributoCSSN:novovalor});

Manipular classes CSS

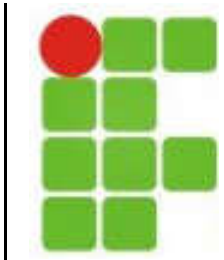


- Para tornar a manipulação CSS mais prática, é possível fazer a alteração CSS de diversos atributos de uma vez, utilizando-se de classes;

- As classes devem estar no devido lugar do código CSS, senão não é possível utilizar este artefato;

- Sim, um elemento poderá possuir mais de uma classe ao mesmo tempo!!!

Manipular classes CSS

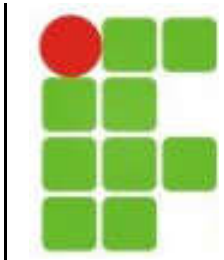


- É possível realizar as seguintes operações:
  - Adicionar uma classe CSS;
  - Remover uma classe CSS;
  - Alternar uma classe CSS.

- OBS: Preste bastante atenção, pois em atributos CSS conflitantes, a preferência será sempre da última classe definida!!!

- **Ex: Se ambas as classes definirem a cor da letra, a cor da letra será o da última classe definida!!!**

Adicionando classe CSS



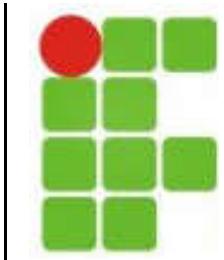
- Adiciona aos elementos selecionados a classe que estará no argumento da ação **addClass**, modificando os estilos presentes em tal classe;

- Para uma classe: SELETOR.**addClass**(classe);

- Para múltiplas classes: SELETOR.**addClass**(classe1 classe2 classeN);

- OBS: Atenção para as situações de conflito!!!

Removendo classe CSS



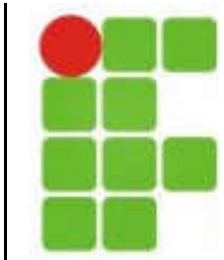
- Remove dos elementos selecionados a classe que estará no argumento da ação **removeClass**, retirnado os estilos presentes em tal classe;

- Para uma classe: SELETOR.**removeClass**(classe);

- Para múltiplas classes: SELETOR.**removeClass**(classe1 classe2 classeN);

- OBS: Atenção para as situações de conflito!!!

Alternando classe CSS



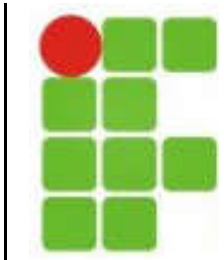
- Adicionará a classe se o elemento não possuir e removerá se ele já possuir, a(s) classe(s) que estiverem no argumento da ação **toggleClass**;

- Para uma classe: SELETOR.**toggleClass**(classe);

- Para múltiplas classes: SELETOR.**toggleClass**(classe1 classe2 classeN);

- OBS: Atenção para as situações de conflito!!!

Atributos customizados

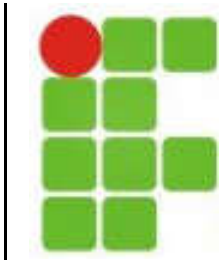


- Esta funcionalidade irá permitir recuperar e alterar valores de atributos customizados, ou seja, criados pelo desenvolvedor;

- É uma forma interessante de guardar valores nos elementos HTML para tomada de decisões, assemelhando-se aos atributos em objetos (POO);

- Para manipular estes atributos, é necessário utilizar a ação data em um seletor.

Atributos customizados

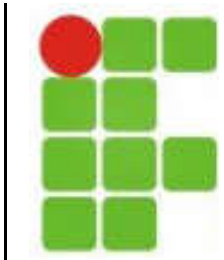


- Qualquer atributo poderá ser manipulado através disto, uma vez que o controle dos nomes dos atributos e de seus valores é de responsabilidade do desenvolvedor;

- Para um atributo ser utilizado, um valor deve ser atribuído a ele inicialmente, uma vez que não são atributos que existem naturalmente com os elementos HTML;

- OBS: Se não for encontrado o atributo customizado, será retornado undefined.

Modificar o valor do atributo customizado



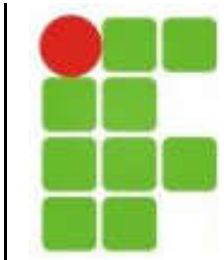
- É necessário alimentar a ação data com o nome do atributo que se deseja modificar o valor e o novo valor para este atributo;

- Se o seletor encontrar diversos elementos HTML, todos serão alterados;

- Sintaxe: SELETOR.**data**(atributoCustomizado,novovalor);

- OBS: Se o atributoCustomizado não estiver presente, ele será inicializado. Caso contrário, será sobrescrito.

Retornar o valor do atributo customizado



- É necessário alimentar a ação **data** com o nome do atributo que se deseja recuperar o valor;
- Se o seletor encontrar diversos elementos HTML, somente o valor do atributo do primeiro elemento encontrado será retornado;
- Utilize o **each** para retornar o de todos;
- Sintaxe: SELETOR.**data**(atributoCustomizado).