XML
Extensible Markup Language

Professor Vinícius Costa

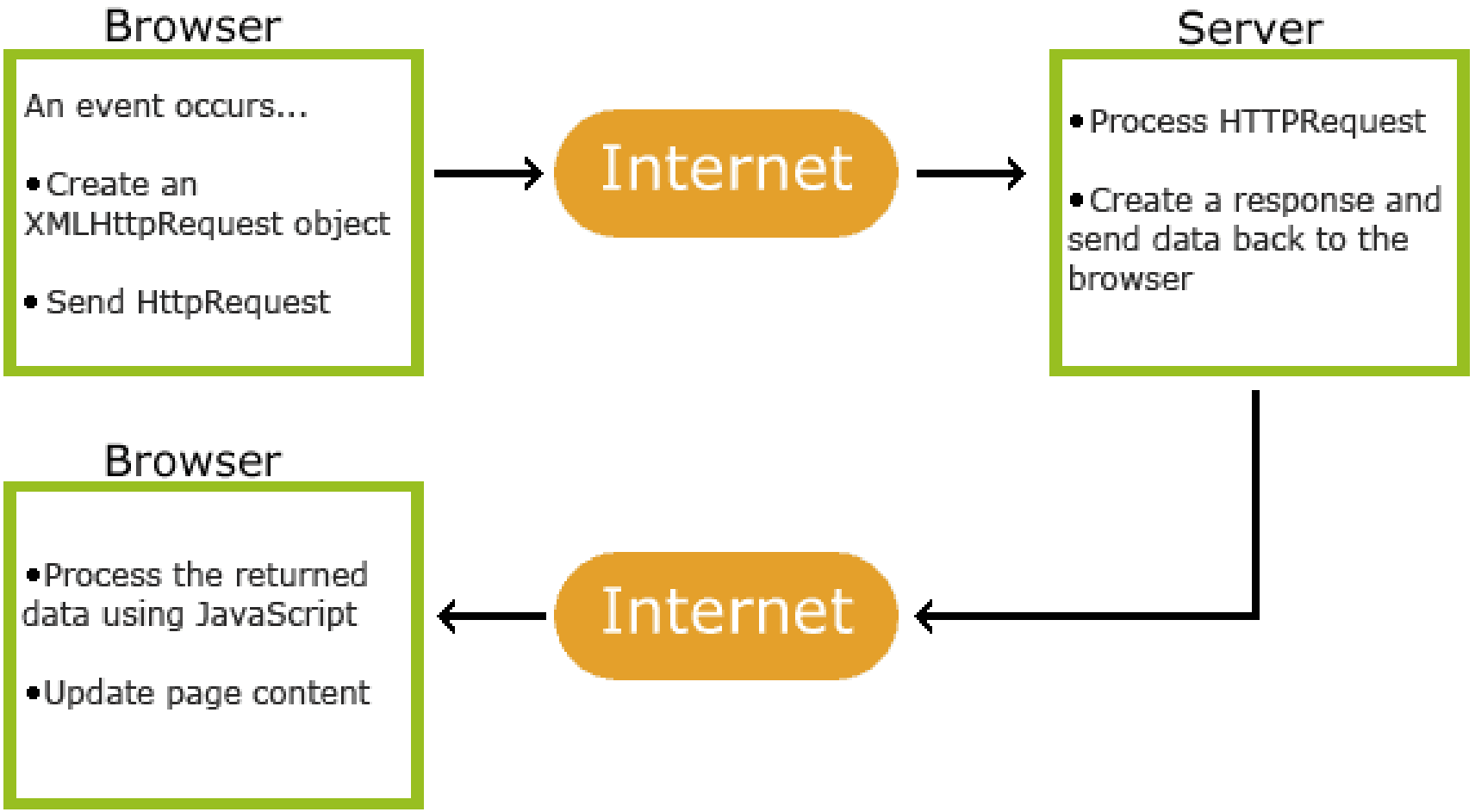
AJAX

- Asynchronous JavaScript And XML
- Não é uma linguagem de programação
- É uma técnica para acessar servidores web através de uma página web
  - Utiliza objeto XMLHttpRequest para solicitar dados do servidor
  - Javascript e DOM HTML para exibir ou tratar os dados

Ajax

- Com Ajax é possível:
  - Atualizar uma página web sem recarrega-la
  - Requisitar dados do servidor mesmo depois que a página já tenha sido carregada
  - Receber dados do servidor mesmo depois que a página já tenha sido carregada
  - Enviar dados para o servidor em segundo plano

Como AJAX Funciona



Como Ajax Funciona

1. Um evento é disparado na página web
2. Um objeto XMLHttpRequest é criado pelo JavaScript
3. O XMLHttpRequest envia uma requisição para o servidor
4. O servidor processa a requisição
5. O servidor envia uma resposta de volta para a página web
6. A resposta é lida pelo JavaScript
7. JavaScript realiza a ação adequada com a resposta, por exemplo, mostrar os dados

Objeto XMLHttpRequest

- O objeto XMLHttpRequest pode ser utilizado para trocar dados com o servidor em segundo plano
- Sendo assim é possível atualizar partes de uma página Web sem recarregar toda a página
- Todos os navegadores modernos suportam XMLHttpRequest

Criando um Objeto XMLHttpRequest

- Sintaxe
  - NomeDaVariavel = new XMLHttpRequest( );

Internet Explorer 6 e Anteriores

- Versões antigas do Internet Explorer utilizam ActiveX ao invés de XMLHttpRequest
- Sintaxe
  - nomeDaVariavel = new ActiveXObject("Microsoft.XMLHTTP");

Ajax Cross Browser

- Verifique se o browser suporta objetos XMLHttpRequest se sim crie tal objeto senão crie um objeto ActiveX
- Exemplo

```
if (window.XMLHttpRequest) {
    xmlhttp = new XMLHttpRequest();
} else {
    xmlhttp = new ActiveXObject("Microsoft.XMLHTTP");
}
```

Metodos do Objeto XMLHttpRequest

Method	Description
new XMLHttpRequest()	Cria um novo objeto XMLHttpRequest
abort()	Cancela a requisição atual
getAllResponseHeaders()	Retorna informações do cabeçalho
getResponseHeader()	Retorna infomações especificas do cabeçalho
open(method,url,async,user, psw)	Especifica a requisição method: tipode requisição GET ou POST url: localização do arquivo async: true (assíncrono) ou false (síncrono) user: nome do usuário (opcional) psw: senha (opcional)
send()	Envia uma requisição para o servidor usando método GET
send(string)	Envia uma requisição para o servidor usando método POST
setRequestHeader()	Adiciona um par rotulo/valor no cabeçalho a ser enviado

Propriedades do Objeto XMLHttpRequest

Property	Description
onreadystatechange	Define uma função que será chamada quando a propriedade readyState mudar
readyState	Mantem o status do XMLHttpRequest. 0: requisição não inicializada 1: conecção com o servidor estabelecida 2: requisição recebida 3: processando requisição 4: requisição finalizada e resposta está pronta
responseText	Retorna os dados da resposta como um texto
responseXML	Retorna os dados da resposta como um XML
status	Retorna o número do status de uma requisição 200: "OK" 403: "Forbidden" 404: "Not Found" Para uma lista completa visite Http Messages Reference
statusText	Retorna o status da requisição em formato texto

Enviando uma Requisição ao Servidor

- Para enviar uma requisição usa-se os metodos open e send do objeto XMLHttpRequest
  - xmlhttp.open("GET","arquivo",true);
  - xmlhttp.send( )

  Onde xhttp é um objeto do tipo XMLHttpRequest

Método GET ou POST

- Get é mais simples e mais rápido que POST
- Use POST nos seguintes casos
  - Quando for enviar uma grande quantidade de dados para o servidor
  - Quando for enviar entradas do usuário
  - Quando for atualizar um arquivo ou banco de dados no servidor

Requisição GET

- Exemplo

```
xmlhttp.open("GET",
   "arquivo?nome=José&sobrenome=Silva", true);
xmlhttp.send();
```

Requisição POST

- Exemplo

```
xmlhttp.open("POST", "arquivo", true);
xmlhttp.setRequestHeader("Content-type", "application/x-www-form-
    urlencoded");
xmlhttp.send("nome=José&sobrenome=Silva");
```

- setRequestHeader(cabeçalho, valor)
  - Adiciona um cabeçalho HTTP para a requisição
  - Cabeçalho: especifica o nome do cabeçalho
  - Valor: especifica o valor do cabeçalho

O Parâmetro URL

  - xhttp.open("GET", "**url**", true);
- O parâmetro url pode ser o endereço de um arquivo ou servidor
- O arquivo pode ser de qualquer tipo, por exemplo, txt, xml, php, java, etc.
- Por questão de segurança não é possível acessar url de outro dominio

O Parâmetro Assíncrono

  - xhttp.open("GET", "url", **true**);
- Indica se a requisição será assíncrona (true) ou síncrona (false)
- Use este parâmetro sempre como true
  - Requisições assíncronas fazem com que o JavaScript não precise esperar pela resposta do servidor
  - Requisições assíncronas permitem que o JavaScript realize outras operações enquanto espera a resposta do servidor

A Propriedade onreadystatechange

- Quando uma requisição recebe uma resposta do servidor é possível executar uma função para tratar esta resposta
- Isso é consequido atribuindo uma função para a propriedade onreadystatechange do objeto XMLHttpRequest

A Propriedade onreadystatechange

- Exemplo

```
function criarAjax(){
  var xhttp = new XMLHttpRequest();
  xhttp.onreadystatechange = mostrarResposta;
  xhttp.open("GET", "arquivo.txt", true);
  xhttp.send();
}

function mostrarResposta() {
    if (this.readyState == 4 && this.status == 200) {
      document.getElementById("paragrafo").innerHTML =
      this.responseText;
    }
}
```

A Propriedade onreadystatechange

- A propriedade readyState mantem o status do XMLHttpRequest
- A propriedade onreadystatechange recebe a função que será executada quando a propriedade readyState mudar
- A propriedade status e statusText guarda o status da requisição feita pelo objeto XMLHttpRequest

Propriedades responseText e responseXML

- responseText
  - Retorna a resposta do servidor como uma string JavaScript
- ResponseXML
  - Retorna a resposta do servidor como um objeto XML DOM

Método getAllResponseHeaders

- Retorna todas as informações de cabeçalho da resposta do servidor
- Exemplo

```
xhttp.onreadystatechange = function() {
  if (this.readyState == 4 && this.status == 200) {
    document.getElementById("demo").innerHTML =
    this.getAllResponseHeaders();
  }
};
```

Método getResponseHeader

- Retorna a informação especificada do cabeçalho da resposta do servidor
- Exemplo

```
xhttp.onreadystatechange = function() {
  if (this.readyState == 4 && this.status == 200) {
    document.getElementById("demo").innerHTML =
    this.getResponseHeader("Last-Modified");
  }
};
```

Exemplo Ajax

```
<!DOCTYPE html>
<html>
<body>
<div id="demo">
  <h2>Let AJAX change this text</h2>
  <button type="button" onclick="loadDoc()">Change Content</button>
</div>
</body>
</html>


function loadDoc() {
          var xhttp = new XMLHttpRequest();
          xhttp.onreadystatechange = function() {
            if (this.readyState == 4 && this.status == 200) {
              document.getElementById("demo").innerHTML = this.responseText;
            }
          };
          xhttp.open("GET", "ajax_info.txt", true);
          xhttp.send();
        }
```