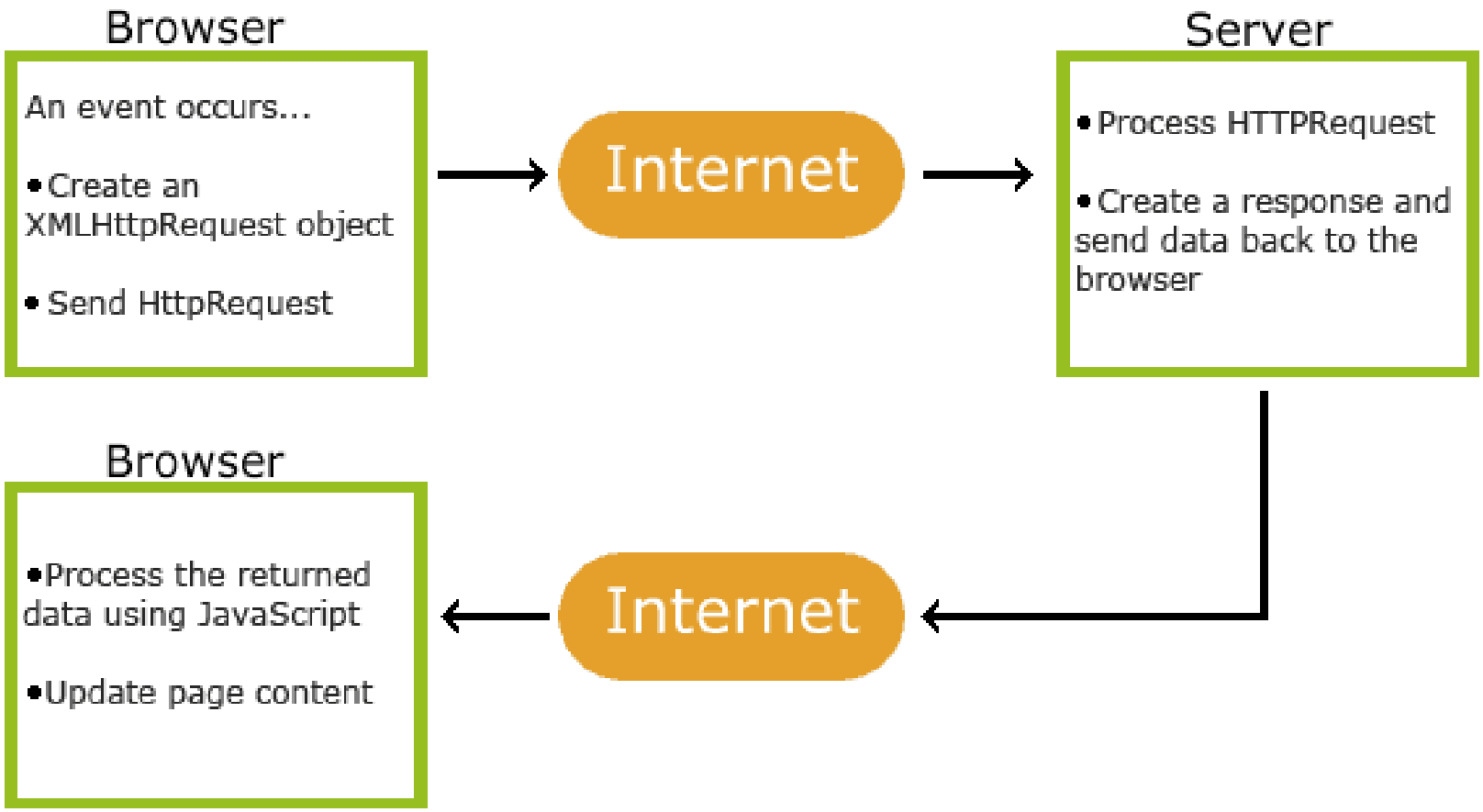
Programação II

PHP com AJAX

AJAX

- AJAX = **A**synchronous **J**avaScript **A**nd **X**ML
- AJAX não é uma linguagem, mas sim uma combinação de:
  - Um objeto **XMLHttpRequest** incorporado ao navegador (para requisitar dados ao servidor);
  - **JavaScript** e **HTML DOM** (para exibir os dados ao usuário).
- AJAX torna possível:
  - buscar dados no servidor, mesmo após o carregamento da página;
  - atualizar o conteúdo da página assincronamente, por meio da manipulação do DOM, sem recarregar a página toda;
  - enviar dados para o servidor em *background*.
- Os dados podem ser transportados em formato XML, *plain text* ou JSON.

AJAX



Fonte: https://www.w3schools.com/js/js_ajax_intro.asp

O objeto XMLHttpRequest

- Todos os navegadores modernos possuem suporte ao objeto **XMLHttpRequest**, utilizado para intercambiar dados entre a página e o servidor web, nos “bastidores”. Partes da página podem então ser atualizadas com os dados retornados pelo servidor, sem que ela seja recarregada totalmente.
- Exemplo de como instanciar um objeto **XMLHttpRequest**:

```
var req = new XMLHttpRequest();
```

- Se for necessário dar suporte a versões antigas do IE (5/6), utilize:

```
if (window.XMLHttpRequest) {
  req = new XMLHttpRequest(); //for modern browsers
} else {
  req = new ActiveXObject("Microsoft.XMLHTTP"); //old IE
}
```

Propriedades de XMLHttpRequest

- onreadystatechange
  - Define uma função que será chamada cada vez que a propriedade **readyState** se altera
- readyState
  - Mantém o status do XMLHttpRequest:
    - 0: requisição não iniciada
    - 1: conexão com o servidor estabelecida
    - 2: requisição recebida
    - 3: processando requisição
    - 4: requisição finalizada e resposta pronta
- responseText
  - Retorna a resposta como texto
- responseXML
  - Retorna a resposta como XML
- status
  - Retorna o número do status HTTP da requisição (200: "OK"; 403: "Forbidden"; 404: "Not Found", etc.)
- statusText
  - Retorna o texto do status (ex: "OK" ; "Not Found")

Principais Métodos de XMLHttpRequest

- open(method, url, async)
  - Especifica os dados de uma requisição
    - method: GET ou POST
    - url: localização do arquivo no servidor
    - async: true (asynchronous) ou false (synchronous)
- send()
  - Envia uma requisição (usado para GET ou POST sem dados)
- send(string)
  - Envia uma requisição (usado para POST com dados)
- setRequestHeader(header, value)
  - Adiciona cabeçalhos HTTP à requisição

Exemplos

- Requisição GET sem dados:

```
req.open("GET", "demo_get.php", true);
req.send();
```

- Requisição GET com dados na URL:

```
req.open("GET", "demo_get.php?fname=Henry&lname=Ford", true);
req.send();
```

- Requisição POST sem dados:

```
req.open("POST", "demo_post.php", true);
req.send();
```

- Requisição POST com envio de dados:

```
req.open("POST", "demo_post.php", true);
req.setRequestHeader("Content-type", "application/x-www-form-urlencoded");
req.send("fname=Henry&lname=Ford");
```

Requisição AJAX

- O arquivo no servidor pode ser de qualquer tipo, como um .txt, .xml ou um *script server side* (como .php ou .asp), que processa algo no servidor antes de enviar a resposta.
- O parâmetro async deve ser sempre true, para que a requisição seja assíncrona.
  - Requisições síncronas estão sendo gradativamente eliminadas dos *web standards*.
  - Assíncrona significa que o JavaScript não terá que ficar esperando pela resposta do servidor; ao invés disso, poderá continuar executando outros scripts enquanto aguarda.
  - A resposta será tratada assim que estiver disponível. Para que isto aconteça, é necessário definir na propriedade onreadystatechange o que acontecerá quando a resposta estiver pronta. Fazemos isso definindo uma função.

Exemplo completo (carrega TXT)

```
<!DOCTYPE html>
<html>
<body>
  <div>
     <h2>Exemplo de AJAX</h2>
     <button type="button" onclick="loadDoc()">Change Content</button>
  </div>
  <div id="demo">
     <p>Este texto será substituído pelo conteúdo do TXT após clicar no botão.</p>
  </div>

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

Crie um arquivo com conteúdo qualquer e salve com este nome

A função é chamada 4 vezes, uma para cada mudança da propriedade readyState

Quando readyState for 4 e o status 200, a resposta está pronta e o GET retornou sucesso

Um elemento da página será atualizado com a resposta do servidor

Exemplo completo (carrega PHP)

```
...
<label>Estado:
    <select name="uf" id="uf" onchange="loadCities()">
      <option value="">Selecione</option>
      <option value="PR">Paraná</option>
      <option value="SC">Santa Catarina</option>
      <option value="RS">Rio Grande do Sul</option>
    </select>
  </label><br><br>
  <label>Cidade:
    <select name="cidade" id="cidade" disabled="disabled">
      <option value="">--- Selecione o estado ---</option>
    </select>
  </label><br><br>
...
```

Exemplo completo (carrega PHP)

```
<script>
function loadCities() {
    var req = new XMLHttpRequest();
    uf = document.getElementById("uf").value;
    req.open("GET", "buscaCidades.php?uf=" + uf, true);
    req.send();
    req.onreadystatechange = function() {
        if (this.readyState == 4 && this.status == 200) {
           if(this.responseText != ""){ // retornou algo
             document.getElementById("cidade").innerHTML = "<option value=\"\">--- Selecione
---</option>";
             document.getElementById("cidade").innerHTML += this.responseText; //resultado do PHP
             document.getElementById("cidade").disabled = ""; // habilita o campo cidade
        }
        else { // vazio
           document.getElementById("cidade").innerHTML = "<option value=\"\">--- Selecione o estado
---</option>";
           document.getElementById("cidade").disabled = "disabled"; // desabilita novamente
        }
      }
   };
}
</script>
```

Exemplo completo (carrega PHP)

```
<?php
// arquivo buscaCidades.php
if(!isset($_GET['uf']) || !in_array($_GET['uf'], array("PR", "SC", "RS"))){
  die(); // tentativa de acessar sem identificar o estado ou com valor inválido
}
else {
   if($_GET['uf'] == "PR"){
      echo "<option value='curitiba'>Curitiba</option>
         <option value='pato branco'>Pato Branco</option>
         <option value='londrina'>Londrina</option>";
  }
  elseif($_GET['uf'] == "SC"){
      echo "<option value='chapeco'>Chapecó</option>
        <option value='guatambu'>Guatambu</option>
        <option value='xaxim'>Xaxim</option>";
  }
  else {
      echo "<option value='poa'>Porto Alegre</option>
        <option value='erechim'>Erechim</option>
        <option value='cerro largo'>Cerro largo</option>";
  }
}
?>
```

Para trazer dados de um BD, substitua pela conexão e consulta SQL, de modo a gerar os <option> dinamicamente.

AJAX

- Mais exemplos:
  - https://www.w3schools.com/php/php_ajax_php.asp
- Os exemplos apresentados empregam o AJAX “puro”, ou seja, trabalhando diretamente com o objeto XMLHttpRequest.
- Existem bibliotecas e frameworks JavasScript que incorporam o recurso AJAX, de modo a facilitar sua utilização.
- A mais famosa sem dúvida é a Jquery
  - https://jquery.com/
- Uma lista mais abrangente pode ser encontrada em:
  - https://en.wikipedia.org/wiki/List_of_Ajax_frameworks