AJAX është një teknikë e zhvillimit web interaktive që përdoret për krijimin e aplikacioneve web. Për të përdorur AJAX duhet të keni njohuri paraprake në JavaScript, HTML, CSS, dhe XML.

AJAX është akronim i **A**synchronous **Ja**vaScript and **X**ML. AJAX është një teknikë e re për krijimin e web aplikacioneve më të mira, më të shpejta. Ajax përdor XHTML për përmbajtjen, CSS për prezantimin, së bashku me Document Object Model dhe JavaScript për shfaqjn dinamike të të dhënave.

- Trasmetim konvencial të të dhënave në web aplikacione duke përdorur dërgimin sinkron. Pra nëse plotësoni një formë në web, dhe klikoni dërgo, ju hapet një faqe e re me informacione të reja nga serveri.
- Me AJAX, kur ju klikoni **submit**, JavaScript do të bëj një kërkesë në server, do të interpretojë rezultatet, dhe do të update-jë përmbajtjen e faqes web menjëherë pa bërë re-load faqes.
- XML zakonisht përdoret si format për marrjen e të dhënave nga server-i.
- AJAX është një teknologji web browser e pavarur nga software i serverit web.
- Një përdorues mund të vazhdojë të përdorë aplikacionin, ndërkohë që programi klient komunikon duke kërkuar informacione në server.
- Ndërveprim intuitiv. Edhe pa klikuar, lëvizja e mouse-it shërben si event.

AJAX bazohet në standartet me burime të hapura

AJAX bazohet në standartet e mëposhtme me burime të hapura:

- Për prezantimin e të dhënave në browser: HTML dhe CSS (Cascading Style Sheets).
- Të dhënat ruhen në formatin XML dhe merren nga server.
- Për marrjen e të dhënave nga browser përdoren objektet XMLHttpRequest.
- JavaScript për të gjitha eventet

AJAX – Suporti Browser

AJAX nuk mund të punojë e pavarur, përdoret në kombinim me teknologjitë e tjera për krijimin e web aplikacioneve interaktive.

JavaScript

- Funksionet JavaScript thërriten kur ndodh një event në faqe.
- Përdoret pothuajse për të gjitha veprimet AJAX.

DOM

- API për aksesimin dhe manipulimin e dokumenteve të strukturuara.
- Riprezanton strukturën e dokumentave të XML dhe HTML.

CSS

- Përdoret për stilimin e dokumentave dhe kontrollohet nga JavaScript.

XMLHttpRequest

- Objekte JavaScript që performojnë veprime asinkrone në server.

Disa shembuj:

Më poshtë paraqitet një listë e web aplikacioneve të famshme që përdorin AJAX.

Google Maps

Një përdorues hartën thjesht duke lëvizur mousin pa klikuar asnjë buton.

- http://maps.google.com/

Google Suggest

Ndërkohë që ju shkruani, Google ofron disa sugjerime, ku mund të përdoren shigjetat për zgjedhjen e rezultateve.

- http://www.google.com/

Gmail

Lista e email-eve update-ohet pa i bërë reload faqes.

- http://gmail.com/

AJAX – Suporti Browser

Jo të gjithë browser-at suportojnë AJAX. Më poshtë paraqitet një listë e plotë e browser-ave kryesorë që suportojnë AJAX.

- **Mozilla Firefox 1.0 dhe mbi 1.0.**
- **Netscape versioni 7.1 dhe mbi 7.1.**
- **Apple Safari 1.2 dhe mbi 1.2.**
- **Microsoft Internet Explorer 5 dhe mbi 5.**
- **Konqueror.**
- **Opera 7.6 dhe mbi 7.6.**

Kur shkruani aplikacionin duhet të merrni në konsideratë browserat që nuk suportojnë AJAX.

Shënim: Kur themi që një browser nuk suporton AJAX, do të thotë që browser-i nuk suporton krijimin e objektit Javascript – objekti XMLHttpRequest.

Kodi specifik për browserat që nuk e suportojnë kodin

Mënyra më e thjeshtë për të bërë kodin të përshtatshëm për një browser është të përdorim blloqet *try...catch* në JavaScript.

AJAX – Suporti Browser

```
<html>
<body>
<script language="javascript" type="text/javascript">
<!--
//Kodi që suporton browser-at
function ajaxFunction(){
 var ajaxRequest;  // Variabla që mundëson Ajax!
try{
   // Opera 8.0+, Firefox, Safari
   ajaxRequest = new XMLHttpRequest();
 }catch (e){
   // Internet Explorer Browsers
   try{
      ajaxRequest = new ActiveXObject("Msxml2.XMLHTTP");
   }catch (e)
      { try{
```

```
         ajaxRequest = new ActiveXObject("Microsoft.XMLHTTP");
      }catch (e){
         // Dicka shkoi gabim
         alert("Brouseri nuk
         funkionon!"); return false;
      }
   }
 }
}
//-->
</script>
<form name='myForm'>
Emri: <input type='text' name='emri' /> <br
/> Koha: <input type='text' name='koha' />
</form>
</body>
</html>
```

Në kodin e mësipërm JavaScript, përpiqemi të krijojmë objektin XMLHttpRequest tre herë.

Hera e parë:

- ajaxRequest = new XMLHttpRequest();

Është për browser Opera 8.0+, Firefox, dhe Safari. Nëse dështon, përpiqet të krijojë objektin edhe dy herë të tjera Internet Explorer:

- ajaxRequest = new ActiveXObject("Msxml2.XMLHTTP");
- ajaxRequest = new ActiveXObject("Microsoft.XMLHTTP");

Hapate veprimeve AJAX

1. dodh një event në anën klient.
2. Krijohet një objekt XMLHttpRequest.
3. Konfigurohet objekti XMLHttpRequest.
4. Objekti XMLHttpRequest realizon një kërkesë asinkrone në Webserver.
5. Webserver-i kthen rezultatin një dokument XML.
6. Objekti XMLHttpRequest thërret funksionin callback() dhe proçeson rezultatin.
7. DOM-i HTML update-ohet.

1. Ndodh një event në anën klient.

- Si rezulat i një eventi thërritet një funksion JavaScript.
- Shembull: Funksioni *validateUserId()* thërritet në një event *onkeyup* në fushën input në form ku id është "userid".
- <input type="text" size="20" id="userid" name="id" onkeyup="validateUserId();">.

2. Krijohet një objekt XMLHttpRequest.

```
var ajaxRequest;  // Variabla që bën të mundur AJAX

function
 ajaxFunction(){ try{

   // Opera 8.0+, Firefox, Safari

   ajaxRequest = new XMLHttpRequest();

 }catch (e){

   // Internet Explorer Browser

   try{

    ajaxRequest = new ActiveXObject("Msxml2.XMLHTTP");

   }catch (e)

      { try{

         ajaxRequest = new ActiveXObject("Microsoft.XMLHTTP");
```

```
      }catch (e){
         // Diçka shkoi gabim
         alert("Browser-i juaj nuk po
         funksionon!");
      }
   }
 }
}
```

3. Konfigurohet objekti XMLHttpRequest.

Në këtë hap, do të shkruajmë një funksion i cili thërritet nga një event klient dhe do të regjistrohet funksioni callback processRequest().

```
function
   validateUserId() {
   ajaxFunction();
   // Me poshte eshte funksioni callback processRequest().
   ajaxRequest.onreadystatechange = processRequest;
   if (!target) target =
   document.getElementById("userid"); var url =
   "validate?id=" + escape(target.value);
   ajaxRequest.open("GET", url, true);
   ajaxRequest.send(null);
}
```

4. Objekti XMLHttpRequest realizon një kërkesë asinkrone në Webserver.

Kodi i mëposhtëm me të zeza është përgjegjës për realizimin e një kërkese në webserver. Kjo është gjithçka realizon kërkesa XMLHttpRequest.

```
function
   validateUserId() {
   ajaxFunction();

   // Me poshte eshte funksioni callback processRequest().

   ajaxRequest.onreadystatechange = processRequest;



   if (!target) target = document.getElementById("emri");
   var url = "validate?id=" + escape(target.value);
   ajaxRequest.open("GET", url, true);
   ajaxRequest.send(null);
```

5. Webserver-i kthen rezultatin një dokument XML.

Ju mund të implementoni skriptin server-side në çdo gjuhë skriptive, megjithatë logjika është si më poshtë:

- Marrja e një kërkese nga një klient.
- Parsimi i inputit nga klienti.
- Realizimi i proçesimit të kërkuar.
- Dërgimi i të dhënave output tek klienti.

Për shembull nëse shkruajmë një servlet, më poshtë paraqitet kodi:

```
public void doGet(HttpServletRequest request, HttpServletResponse response)
   throws IOException, ServletException
{
   String targetId = request.getParameter("id");
   if ((targetId != null) && !accounts.containsKey(targetId.trim()))
   {
      response.setContentType("text/xml");
      response.setHeader("Cache-Control", "no-
      cache");
      response.getWriter().write("true");
   }
   else
   { response.setContentType("text/xml"); response.setHeader("Cache-
   Control", "no-cache"); response.getWriter().write("false");
   }
}
```

6. Objekti XMLHttpRequest thërret funksionin callback() dhe proçeson rezultatin.

Objekti XMLHttpRequest konfigurohet që të thërrasë funksionin processRequest() ku është një ndryshim gjendjeje në *readyState* të objektit *XMLHttpRequest*.Më pas ky funksion do të marrë rezulatatin nga server dhe do të realizojë proçesimin e kërkuar. Siç tregohet në shembullin më poshtë, vendos një mesazh variabël në true ose false bazuar në vlerën e kthyer nga Webserver.

```
function processRequest() {
   if (req.readyState == 4) {
      if (req.status == 200) {
         var message = ...;
...
}
```

7. DOM-i HTML update-ohet

Ky është hapi i fundit, faqja web HTML do të update-ohet. Ndodh sipas hapave të mëposhtme:

- JavaScript merr një referencë tek secili element apo tag në një faqe web duke përdorur DOM API.
- Mënyra më e mirë është duke e thërritur elementin.

```
document.getElementById("MesazhiPerdoruesit"),
// ku "MesazhiPerdoruesit" eshte atributi ID
// i nje tagu ne HTML
```

- JavaScript përdoret për modifikimin e atributeve të tageve; modifikimin e stilit të karaktersirikave; ose të shtojë, zhvendosë, ose modifikojë taget fëmijë. Më poshtë paraqitet një shembull:

```
<script type="text/javascript">
<!--
 function
    setMessageUsingDOM(mesazhi) {
    var MesazhiPerdoruesitTag =
     document.getElementById("MesazhiPerdoruesit");
      var mesazhiTekst;
    if (mesazhi == "false") {
       MesazhiPerdoruesitTag.style.color =
       "red"; mesazhiTekst = "Id e
       gabuar";
    } else {
       MesazhiPerdoruesitTag.style.color =
       "green"; mesazhiTekst = "Id e sakte";
    }
    var mesazhiTrupit = document.createTextNode(mesazhiTekst);



    // nese krijohet mesazhiTrupit
    // zhvendose ate perndryshe kontrollon nje element te ri
    if (MesazhiPerdoruesitTag.childNodes[0]) {
    MesazhiPerdoruesitTag.replaceChild(mesazhiTrupit,
    MesazhiPerdoruesitTag.childNodes[0]);
} else {
    MesazhiPerdoruesitTag.appendChild(mesazhiTrupit);
}
    }
    -->
    </script>
    <body>
    <div id=" MesazhiPerdoruesit"><div>
    </body>
```