16-5 Describe each of the following terms in the context of Ajax:  
  
a. type-ahead: can use an element’s keyup event handler to make asynchronous requests.

b. edit-in-place: a type of functionality which allows users to “edit” various portions of a web page directly in the browser. Sending each entry asynchronously allows the user to address invalid entries quickly, rather than making edits and resubmitting the entire form repeatedly until all entries are valid.

c. partial page update: helps make web applications more responsive, making them feel more like desktop applications.

d. asynchronous request: a request that allows the user to interact with the client while the request is being processed.

e. XMLHttpRequest: the Ajax component that manages interaction with the server, commonly abbreviated as XHR.

f. “raw” Ajax: uses Javascript to send asynchronous requests to the server, then updates the page using the DOM.

g. callback function: the event handler for the XMLHttpREquest object’s readystatechange event. Updates only a designated part of the page.

h. same origin policy: the web application and its resources must reside on the same web server.

i. Ajax libraries: Ajax toolkits such as jQuery, ASP.NET Ajax and JSF’s ajax. They provide powerful ready-to-use controls and functions that enrich web applications and simplify JavaScript coding making it cross-browser compatible.

j. RIA: Web applications that approximate the look, feel and usability of desktop applications. Typically improve performance and rich GUI.

16.6

16-6.js

Text

Description automatically generatedA picture containing text

Description automatically generated

*var* asyncRequest; // variable to hold XMLHttpRequest object

// set up and send the asynchronous request to get the XML file

*function* getImages(*url*) {

    // attempt to create XMLHttpRequest object and make the request

    try {

        asyncRequest = new *XMLHttpRequest*(); // create request object

        // register event handler

        asyncRequest.addEventListener(

            "readystatechange", processResponse, false);

        asyncRequest.open("GET", *url*, true); // prepare the request

        asyncRequest.send(null); // send the request

    } // end try

    catch (exception) {

        alert("Request Failed");

    } // end catch

} // end function getImages

// parses the XML response; dynamically creates an undordered list and

// populates it with the response data; displays the list on the page

*function* processResponse() {

    // if request completed successfully and responseXML is non-null

    if (asyncRequest.readyState == 4 && asyncRequest.status == 200 &&

        asyncRequest.responseXML) {

        clearImages(); // prepare to display a new set of images

        // get the covers from the responseXML

*var* covers = asyncRequest.responseXML.getElementsByTagName(

            "cover")

        // get base URL for the images

*var* baseUrl = asyncRequest.responseXML.getElementsByTagName(

            "baseurl").item(0).firstChild.nodeValue;

        // get the placeholder div element named covers

*var* output = document.getElementById("covers");

        // create an unordered list to display the images

*var* imagesUL = document.createElement("ul");

        // place images in unordered list

        for (*var* i = 0; i < covers.length; ++i) {

*var* cover = covers.item(i); // get a cover from covers array

            // get the image filename

*var* image = cover.getElementsByTagName("image").

                item(0).firstChild.nodeValue;

            // create li and img element to display the image

*var* imageLI = document.createElement("li");

*var* imageTag = document.createElement("img");

*var* imageTitle = document.createElement("h2");

            // set img element's src attribute

            imageTag.setAttribute("src", baseUrl + escape(image));

            imageLI.appendChild(imageTag); // place img in li

            imageLI.appendChild(imageTitle);

            imagesUL.appendChild(imageLI); // place li in ul

            imageTitle.style.display="none";

            title = cover.getElementsByTagName("title").

                item(0).firstChild.nodeValue;

            imageTitle.innerHTML = title;

            imageTag.addEventListener("mouseover",showTitle,false);

            imageTag.addEventListener("mouseout",showTitle,false);

        } // end for statement

        output.appendChild(imagesUL); // append ul to covers div

    } // end if

} // end function processResponse

*function* showTitle(*e*){

*var* titleDisplayStyleProperty = *e*.target.nextSibling.style.display;

    if(titleDisplayStyleProperty=="none")

*e*.target.nextSibling.style.display = "";

    else

*e*.target.nextSibling.style.display = "none";

}

*function* hideTitle(*e*){

*e*.target.nextSibling.style.display = "none";

}

// clears the covers div

*function* clearImages() {

*var* covers = document.getElementById("covers");

    removeAllChildNodes(covers);

} // end function clearImages

*function* removeAllChildNodes(*parent*) {

    while (*parent*.firstChild) {

*parent*.removeChild(*parent*.firstChild);

    }

}

// register event listeners

*function* registerListeners() {

    document.getElementById("all").addEventListener(

        "click", *function* () { getImages("all.php"); }, false);

    document.getElementById("simply").addEventListener(

        "click", *function* () { getImages("simply.xml"); }, false);

    document.getElementById("howto").addEventListener(

        "click", *function* () { getImages("howto.xml"); }, false);

    document.getElementById("dotnet").addEventListener(

        "click", *function* () { getImages("dotnet.xml"); }, false);

    document.getElementById("javaccpp").addEventListener(

        "click", *function* () { getImages("javaccpp.xml"); }, false);

    document.getElementById("none").addEventListener(

        "click", clearImages, false);

} // end function registerListeners

window.addEventListener("load", registerListeners, false);

all.xml

<?xml version="1.0" encoding="UTF-8"?>

<covers>

    <baseurl>http://localhost/</baseurl>

    <cover>

        <image>vb2010htp.jpg</image>

        <title>Visual Basic How To Program</title>

    </cover>

    <cover>

        <image>vcsharp20120htp.jpg</image>

        <title>Visual C# How To Program</title>

    </cover>

    <cover>

        <image>simplyvb2008.jpg</image>

        <title>Simply VB 2008</title>

    </cover>

</covers>

dotnet.xml

<?xml version="1.0" encoding="UTF-8"?>

<covers>

    <baseurl>http://localhost/</baseurl>

    <cover>

        <image>vb2010htp.jpg</image>

        <title>Visual Basic How To Program</title>

    </cover>

    <cover>

        <image>vcsharp2010htp.jpg</image>

        <title>Visual C# How To Program</title>

    </cover>

    <cover>

        <image>simplyvb2008.jpg</image>

        <title>Simply VB 2008</title>

    </cover>

</covers>

javacpp.xml

<?xml version="1.0" encoding="UTF-8"?>

<covers>

    <baseurl>http://localhost/</baseurl>

    <cover>

        <image>java.jpg</image>

        <title>How To Program in Java</title>

    </cover>

</covers>

simply.xml

<?xml version="1.0" encoding="UTF-8"?>

<covers>

    <baseurl>http://localhost/</baseurl>

    <cover>

        <image>simplycpp.jpg</image>

        <title>Simply C++</title>

    </cover>

    <cover>

        <image>simplyvb2008.jpg</image>

        <title>Simply VB 2008</title>

    </cover>

    <cover>

        <image>simplyjava.jpg</image>

        <title>Simply Java</title>

    </cover>

</covers>

howto.xml

<?xml version="1.0" encoding="UTF-8"?>

<covers>

    <baseurl>http://localhost/</baseurl>

    <cover>

        <image>cpphtp8.jpg</image>

        <title>C++ How to Program</title>

    </cover>

    <cover>

        <image>jhtp9.jpg</image>

        <title>Java How to Program</title>

    </cover>

    <cover>

        <image>vb2010htp.jpg</image>

        <title>Visual Basic How To Program</title>

    </cover>

    <cover>

        <image>vcsharp2010htp.jpg</image>

        <title>Visual C# How To Program</title>

    </cover>

</covers>

Javacpp.xml

<?xml version="1.0" encoding="UTF-8"?>

<covers>

    <baseurl>http://localhost/</baseurl>

    <cover>

        <image>java.jpg</image>

        <title>How To Program in Java</title>

    </cover>

</covers>

Simply.xml

<?xml version="1.0" encoding="UTF-8"?>

<covers>

    <baseurl>http://localhost/</baseurl>

    <cover>

        <image>simplycpp.jpg</image>

        <title>Simply C++</title>

    </cover>

    <cover>

        <image>simplyvb2008.jpg</image>

        <title>Simply VB 2008</title>

    </cover>

    <cover>

        <image>simplyjava.jpg</image>

        <title>Simply Java</title>

    </cover>

</covers>

Index.php

<!DOCTYPE html>

<!-- Fig. 16.8: PullImagesOntoPage.html -->

<!-- Image catalog that uses Ajax to request XML data asynchronously. -->

<html>

<head>

<meta charset="utf-8">

<title> Pulling Images onto the Page </title>

<style type = "text/css">

li { *display*: inline-block; *padding*: 4px; *width*: 120px; }

img { *border*: 1px solid black }

</style>

</head>

<body>

<input type = "radio" name ="Books" value = "all"

id = "all"> All Books

<input type = "radio" name = "Books" value = "simply"

id = "simply"> Simply Books

<input type = "radio" name = "Books" value = "howto"

id = "howto"> How to Program Books

<input type = "radio" name = "Books" value = "dotnet"

id = "dotnet"> .NET Books

<input type = "radio" name = "Books" value = "javaccpp"

id = "javaccpp"> Java/C/C++ Books

<input type = "radio" checked name = "Books" value = "none"

id = "none"> None

<div id = "covers"></div>

</body>

</html>

<script src="16-6.js"></script>

Router.php

<?php

// The file test.xml contains an XML document with a root element

// and at least an element /[root]/title.

header('Access-Control-Allow-Origin: \*');

header("Access-Control-Allow-Headers: X-API-KEY, Origin, X-Requested-With, Content-Type, Accept, Access-Control-Request-Method");

header("Access-Control-Allow-Methods: GET, POST, OPTIONS, PUT, DELETE");

header('Content-type: text/xml');

$file = strval($\_GET["file"]);

if (file\_exists($file)) {

    $xml = file\_get\_contents($file);

    echo $xml;

} else {

    exit('Failed to open test.xml.');

}

?>

16-7

Graphical user interface, text, application, email

Description automatically generatedGraphical user interface, text, application, email

Description automatically generated

Index.php

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>More Forms</title>

</head>

<body onload="main()">

    <h1>Feedback Form</h1>

    <p>Please fill out this form to help us improve our site</p>

    <form action="post">

        <p>

            <input type="hidden" vlaue="dietel@deitel.com"/>

            <input type="hidden" name="subject" vlaue="Feedback Form"/>

            <input type="hidden" name="redirect" value="main.html" />

        </p>

        <p><label for="Name">Name:

            <input id="name" name="name" type="text" size="25"/>

        </label></p>

        <!-- <textarea> creates a multiline textbox -->

        <p><label for="comments">Comments:<br/>

            <textarea id="comments" name="comments" cols="36" rows="4"></textarea>

        </label></p>

        <p><label for="email">E-mail Address:

            <input id="email" type="password" name="email" size="25"/>

        </label></p>

        <p>

            <strong>Things you liked:</strong><br/>

            <label>Site design

                <input id="thingsliked" type="checkbox" name="thingsliked" value="Design"/>

            </label>

            <label>Ease of use

                <input type="checkbox" name="thingsliked" value="Ease"/>

            </label>

            <label>Images

                <input type="checkbox" name="thingsliked" value="Images"/>

            </label>

            <label>Source code

                <input type="checkbox" name="thingsliked" value="Code"/>

            </label>

        </p>

        <p>

            <strong>How did you get to our site?:</strong><br/>

            <label>Search engine

                <input name="howtosite" type="radio" value="search engine" checked="checked"/>

            </label>

            <label>Links from another site

                <input name="howtosite" type="radio" value="link" checked="checked"/>

            </label>

            <label>Reference in a book

                <input name="howtosite" type="radio" value="book"/>

            </label>

            <label>Other

                <input name="howtosite" type="radio" value="other"/>

            </label>

        </p>

        <p>

            <label>Rate our site:

                <select name="rating">

                    <option selected="selected">Amazing</option>

                    <option>10</option>

                    <option>9</option>

                    <option>8</option>

                    <option>7</option>

                    <option>6</option>

                    <option>5</option>

                    <option>4</option>

                    <option>3</option>

                    <option>2</option>

                    <option>1</option>

                    <option>Awful</option>

                </select>

            </label>

        </p>

        <p>

            <input type="submit" value="Submit"/>

            <input type="reset" value="Clear"/>

        </p>

    </form>

</body>

</html>

<script src="16.7.js"></script>

16-7.js

//global variables

*var* element;

*var* email;

*var* asyncRequest;

// initial entry point.

*function* main(){

*var* name        = document.getElementById("name")

*var* email       = document.getElementById("email");

*var* comments    = document.getElementById("comments");

*var* thingsLiked = document.getElementById("thingsliked");

    // check for focus event

    name.addEventListener("focus", ensureNotEmpty);

    email.addEventListener("focus",ensureNotEmpty);

    comments.addEventListener("focus",ensureNotEmpty);

    thingsLiked.addEventListener("focus",ensureNotEmpty);

    // check for blur event

    name.addEventListener("blur", ensureNotEmpty);

    email.addEventListener("blur",ensureNotEmpty);

    comments.addEventListener("blur",ensureNotEmpty);

    thingsLiked.addEventListener("blur",ensureNotEmpty);

}

*function* ensureNotEmpty(*event*){

*var* selectedElement = document.getElementById(event.srcElement.id);

    if(event.type == "focus"){

        element = document.getElementById(event.srcElement.id);

    }

    if(event.type == "blur" && element == selectedElement){

        if(selectedElement.value){

            removeWarning(selectedElement);

        }

        else{

            if(warningDoesNotExist(element)){

                createWarning(element);

            }

        }

        if(selectedElement.id=="email"){

            checkEmailFormat(selectedElement);

            checkEmailList(selectedElement);

        }

    }

}

*function* createWarning(*element*){

*var* id = element.id+"-warning";

*var* label   = document.createElement("label");

    label.innerHTML = " |Please do not leave blank|";

    label.id    = id;

*element*.parentNode.appendChild(label);

}

*function* removeWarning(*element*){

*var* id = *element*.id+"-warning";

    warning = document.getElementById(id);

    if(warning && *element*.value)

        warning.remove();

}

*function* warningDoesNotExist(*element*){

*var* id = *element*.id+"-warning";

    return !document.getElementById(id);

}

*function* checkEmailFormat(*emailElement*){

*var* pattern = /^[a-zA-Z0-9.!#$%&'\*+/=?^\_`{|}~-]+@[a-zA-Z0-9-]+(?:\.[a-zA-Z0-9-]+)\*$/;

    if(pattern.test(*emailElement*.value)){

        emailWarning = document.getElementById("email-format-warning");

        if(emailWarning)

*emailElement*.parentNode.removeChild(*emailElement*);

    }

    else{

        addEmailWarning(*emailElement*, " |Invalid email|");

    }

}

*function* addEmailWarning(*emailElement*, *warning*){

    if(!document.getElementById("email-format-warning")){

*var* label   = document.createElement("label");

        label.innerHTML = *warning*;

        label.id    = "email-format-warning";

*emailElement*.parentNode.appendChild(label);

    }

}

*function* addBlackListEmailWarning(*emailElement*, *warning*){

    if(!document.getElementById("email-blacklist-warning")){

*var* label   = document.createElement("label");

        label.innerHTML = *warning*;

        label.id    = "email-blacklist-warning";

*emailElement*.parentNode.appendChild(label);

    }

}

*function* checkEmailList(*emailElement*){

    email = *emailElement*;

    asyncRequest = new *XMLHttpRequest*();

    asyncRequest.addEventListener("readystatechange",processResponse, false);

    asyncRequest.open("GET","http://localhost/16-7.php",true);

    asyncRequest.send(null);

}

*function* processResponse(){

    if(asyncRequest.readyState==4 && asyncRequest.status==200 && asyncRequest.responseXML){

*var* emails = asyncRequest.responseXML.getElementsByTagName("email");

*var* removeWarning = false;

        for(*var* i=0; i < emails.length;i++){

            removeWarning = true;

            if(email.value==emails[i].textContent){

                addBlackListEmailWarning(email, " |black listed|");

                removeWarning = false;

            }

        }

        if(removeWarning && document.getElementById("email-blacklist-warning")){

            document.getElementById("email-blacklist-warning").remove();

        }

    }

}

16-7.php

<?php

// The file test.xml contains an XML document with a root element

// and at least an element /[root]/title.

header('Access-Control-Allow-Origin: \*');

header("Access-Control-Allow-Headers: X-API-KEY, Origin, X-Requested-With, Content-Type, Accept, Access-Control-Request-Method");

header("Access-Control-Allow-Methods: GET, POST, OPTIONS, PUT, DELETE");

header('Content-type: text/xml');

$file = "16-7.xml";

if (file\_exists($file)) {

    $xml = file\_get\_contents($file);

    echo $xml;

} else {

    exit('Failed to open test.xml.');

}

?>

16-7.xml

<?xml version="1.0" encoding="UTF-8"?>

<email>blackhat@aol.com</email>

16-8

Graphical user interface, text, application, email

Description automatically generatedGraphical user interface, text, application

Description automatically generated

16-8.php

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <link rel="stylesheet" href="16-8.css">

    <title>Product Catalog</title>

</head>

<body onload="main()">

    <table id="table">

        <thead>

            <th>Product</th>

            <th>Image</th>

            <th>Description</th>

        </thead>

        <tbody id="tbody">

        </tbody>

    </table>

</body>

</html>

<script src="16-8.js"></script>

Router.php

<?php

header('Content-Type: text/json;charset=utf-8');

$file = $\_GET["file"];

$data = file\_get\_contents($file);

if($data)

    echo $data;

descriptions.json

{

*"descriptions"*:[

        {

*"product"*:"professional pencil",

*"description"*:"The most verstile tool any programmer can have. With this professional pencil you'll be able to sketch out plans and fix mistakes!"

        },

        {

*"product"*:"coffee mug",

*"description"*:"Keep your programming skills sharp and your coffee hot with this one of a kind coffee mug."

        },

        {

*"product"*:"programming book",

*"description"*:"Learn how to program effectively by reading this book."

        }

    ]

}

Summary.json

{

*"products"*:[

        {

*"product"*:"professional pencil",

*"image"*:"pencil.jpg",

*"description"*:"The most verstile tool any programmer can have. With this professional pencil you'll be able to sketch out plans and fix mistakes!"

        },

        {

*"product"*:"coffee mug",

*"image"*:"coffee\_mug.jpg",

*"description"*:"Keep your programming skills sharp and your coffee hot with this one of a kind coffee mug."

        },

        {

*"product"*:"programming book",

*"image"*:"programming\_book.jpg",

*"description"*:"Learn how to program effectively by reading this book."

        }

    ]

}

16-8.js

// initial entry point

*function* main(){

    initializeCatalog();

    setImageHandlers();

}

*function* setImageHandlers(){

*var* images = document.getElementsByClassName("thumbnail");

    for(*var* i=0; i <images.length; i++){

        images[i].addEventListener("mouseover",enlargeImageSize);

    }

}

*function* setButtonHandlers(){

*var* buttons = document.getElementsByTagName("input");

    for(*var* i=0; i<buttons.length; i++){

        buttons[i].addEventListener("click", showDescription);

    }

}

*function* showDescription(*e*){

    getDescription(*e*.srcElement.id);

}

*var* gProduct;

*function* getDescription(*product*){

    gProduct = *product*;

    asyncRequest = new *XMLHttpRequest*();

    asyncRequest.addEventListener("readystatechange",toggleDescription, false);

    asyncRequest.open("GET","http://localhost/router.php?file=descriptions.json",true);

    asyncRequest.send(null);

}

*function* toggleDescription() {

    if(asyncRequest.readyState==4 && asyncRequest.status==200){

*var* products = JSON.parse(asyncRequest.response).descriptions;

        for(*var* i=0; i < products.length; i++){

            if(gProduct == products[i].product){

*var* productDescription = document.getElementById(gProduct+"-desc")

                if(!productDescription.classList.contains("displayed")){

                    productDescription.innerHTML = products[i].description;

                    productDescription.classList.add("displayed");

                }

                else{

                    productDescription.classList.remove("displayed");

                    productDescription.innerHTML = "";

                }

            }

        }

    }

}

*function* enlargeImageSize(*e*){

    if(*e*.srcElement.className=="thumbnail")

*e*.srcElement.className = "product";

}

*function* decreaseImageSize(*image*){

    if(*image*.className=="product")

*image*.className = "thumbnail";

}

*function* initializeCatalog(){

    asyncRequest = new *XMLHttpRequest*();

    asyncRequest.addEventListener("readystatechange", generateProductCatalog, false);

    asyncRequest.open("GET","http://localhost/router.php?file=summary.json",true);

    asyncRequest.send(null);

}

*function* generateProductCatalog(){

    if(asyncRequest.readyState==4 && asyncRequest.status==200){

*var* tbody = document.getElementById("tbody");

*var* products = JSON.parse(asyncRequest.response).products;

        for(*var* i=0; i < products.length; i++){

*var* row =   "<td><input type='button' id='"+products[i].product+"' value='"+products[i].product+"'/></td>"+

                        "<td><img onmouseout='decreaseImageSize(this)' class='thumbnail' src="+products[i].image+" alt=''></td>"+

                        "<td id='"+products[i].product+"-desc'></td>"

*var* newRow = tbody.insertRow();

            newRow.innerHTML=row;

        }

    }

    setImageHandlers();

    setButtonHandlers();

}