Web Advanced: Javascript APIs

"We will learn JavaScript properly. Then, we will learn useful design patterns. Then we will pick up useful tools for making cool things better."

FALL 2018

SESSION #7

FORMS & FORM EVENTS

HANDLING AJAX EVENTS

jaink@newschool.edu

https://canvas.newschool.edu/courses/1407281

https://classroom.github.com/classrooms/42809645-parsons-web-advanced-javascript-fall-2018

RECAP

DOM FOR FORMS

- ➔ The standard way to interact with a page.
- ➔ Form Elements
- ➔ Form Properties & Methods
- ➔ Form Events
- ➔ Validation

THE FORM STRUCTURE

```
  <form name="search" action="search"
id="searchForm" method="GET">
    <input type="text" name="searchBox">
    <button type="submit">Search</button>
  </form>
```

https://repl.it/@jaink/webjavascript18-6

THE FORM ELEMENTS

```
 <form name="search" action="search" id="searchForm">

    <label for="SearchBox">Type your search term
here:</label>
    <input type="text" name="searchBox" id="SearchBox">

    <label for="searchType">Select your search
type:</label>
    <select name="searchType" id="searchType">
        <option value="quick">Quick Search</option>
        <option value="detail">Detailed Search</option>
    </select>

      <label for="searchArea">Fields to search
in:</label>
      <label><input type="checkbox" name="searchArea"
value="local" required-field>Local Sites</label>
      <label><input type="checkbox" name="searchArea"
value="wide" required-field>Wide Sites</label>
      <label><input type="checkbox" name="searchArea"
value="global" required-field>Global Sites</label>


    <label>Select your search location:</label>
    <label><input type="radio" name="searchScope"
value="1">Inside Site</label>
    <label><input type="radio" name="searchScope"
value="2">Outside Site</label>

      <label for="SearchPass">What's your age:</label>
      <input type="number" name="searchAge"
id="SearchAge" required-field>
```

THE DOM FOR FORMS

```
const form = document.forms[0];

const form =
document.getElementById('searchForm')

const form = document.forms["search"];

const form = document.forms.search;



const inp =
document.getElementsByName("searchBox");

const inp = form.elements[0];

const inp = form.elements.searchBox;

const inp = form["searchBox"];


const inp_attribute = inp.type;
```

FORM METHODS

```
form.submit(); //submits the form
form.reset(); //resets all input data
```

FORM EVENTS

```
const inp = form.elements.searchBox;


inp.addEventListener('focus', function(){
console.log("focused on",inp.name)});

inp.addEventListener('blur', function(){
console.log("leaving",inp.name)});

inp.addEventListener('change', function(){
console.log("leaving after changing",inp.name)});


form.addEventListener('submit', function(){

    console.log("submitting form",this.name);

});
```

FORM DATA

Get value:

```
const inp = form.elements.searchBox;

inp.addEventListener('change', function(){
    console.log("leaving after
changing",this.value);
});
```

Set value:

```
const inp = form.elements.searchBox;
inp.value = "Enter your search term here";
```

Get/Set Attributes:

```
inp.getAttribute("required")
inp.hasAttribute("required")
inp.required = 1;
```

FORM DATA

Password field values:

```
let input_pass = form.elements.SearchPass.value;
```

Checkbox values:

```
let input_area = form.elements.searchArea;
const form_area_vals = [];
for (i=0; i < input_area.length; i++) {
  if (input_area[i].checked) {
    form_area_vals.push(input_area[i].value);
}};


const form_area_vals = Array.from(input_area)
.filter(function(input_area) {
  return input_area.checked
})
.map( function(item) {
    return item.value
})

const form_area_vals = [...input_area].filter(
(input_area) =>
   input_area.checked ).map( (item) => item.value
)
```

FORM DATA

Radio button values:

```
var input_scope = form.elements.searchScope;
var form_scope_vals = [];
for (var i=0; i < input_scope.length; i++) {
  if (input_scope[i].checked) {
    form_scope_vals.push(input_scope[i].value);
}};
```

Dropdown values:

```
var input_type = form.elements.searchType.value;
```

FORM SUBMISSIONS

- ➔ HTTP clients and servers.
- ➔ Client: Javascript, HTML ...
- ➔ Server: Apache, Nginx, PHP, Ruby ...
- ➔ HTTP Requests and transfers

AJAX

- ➔ Allows Javascript to take over the form submissions and process outside the user interface, transmitting the data asynchronously.
- ➔ Asynchronous
- ➔ Javascript
- ➔ XML (though mostly JSON now - so perhaps we should call it AJAJ?)

AJAX DOM

AJAX uses the XMLHttpRequest (XHR) DOM object.
It can build HTTP requests, send them, and retrieve their results .

```
cont xhr = new XMLHttpRequest();

xhr.open("GET", "http://google.com/search",
true); //3rd option is for async or sync

xhr.send("{'image_id':1}");
```

http://devdocs.io/dom/xmlhttprequest

AJAX PROPERTIES

readyState:

0: The object has been created, but the open() method hasn't been called.
1: The open() method has been called, but the request hasn't been sent.
2: The request has been sent; headers and status are received and available.
3: A response has been received from the server.
4: The requested data has been fully received.

status:

200, 201 for a successful request,
404 if it cannot find the endpoint

https://developer.mozilla.org/en-US/docs/Web/HTTP/Status

AJAX PROPERTIES

response: returns the response sent back from the server

responseType: returns the type of data contained in the response

responseText: returns the text version of the response

AJAX METHODS

```
xhr.setRequestHeader(); //sets the request data type:
text, JSON etc.
eg. xhr.setRequestHeader("Content-Type",
"application/json");

xhr.onreadystatechange = processXresponse;
function processXresponse() {
    if (xhr.readyState === 4 && xhr.status === 200) {
        // completed - do something with the response
    }
    if (xhr.readyState === 4 && xhr.status !== 200) {
        // error - respond accordingly
    }
}

xhr.onload = callback; //same as xhr.readyState = 4
```

AJAX METHODS - GET

```
let button = document.getElementById('GetUsers');
button.addEventListener("click", getUserData);
function getUserData() {
    var url = "https://reqres.in/api/users";
    var xhr = new XMLHttpRequest();
    xhr.onreadystatechange = function() {
        if (xhr.readyState === 4 ) {
            if (xhr.status === 200) {

document.getElementById("Output").innerHTML =
xhr.responseText;
            } else {

document.getElementById("Output").innerHTML = "There was
an error";
            }
        }
    }
    xhr.open("GET", url, true);
    xhr.send(null);
}
```

AJAX METHODS - POST

```
button.addEventListener("click", sendUserData);
function sendUserData() {
    var url = "https://reqres.in/api/users";
    var xhr = new XMLHttpRequest();
    xhr.onreadystatechange = function() {
        if (xhr.readyState === 4 ) {
            if (xhr.status === 200) {

document.getElementById("Output").innerHTML =
xhr.responseText;
            } else {

document.getElementById("Output").innerHTML = "There was
an error";
            }
        }
    }
    xhr.open("POST", url, true);
    xhr.send("name=jason"); //data needs to be the
format expected, name=value pairs, json etc.
}
```

FORMDATA

Collects all data in a form in an object to be sent with AJAX:

```
var data = new FormData(form);
xhr.send(data);
```

To append:

```
data.append("name","name of person");
```

To loop:

```
for (let pair of data.entries()) {
     jsonObject[pair[0]] = pair[1];
}
```

To send:

```
xhr.send(data);
```

JSON OVERVIEW

JSON is a string representation of the object literal notation:

```
var person = {
            "first_name": "John",
            "last_name": "Smith",
            "Schools": ["Parsons", "NYU"]
        };
```

Convert to string to pass as parameters:

```
var person_payload = JSON.stringify(person)
//
"{"firstName":"John","lastName":"Doe","age":30}"
```

Convert to string to pass as parameters:

```
var person_object = JSON.parse(person_payload)

// {firstName: "John", lastName: "Doe", age: 30}
```

FETCH API

Fetch uses a concept in Javascript called Promise to logically chain asynchronous responses together.

```
button.addEventListener("click", getUserData);

function getUserData() {
  let url = "https://reqres.in/api/users";
    let xhr = new XMLHttpRequest();
    fetch(url)
      .then(function(response) {
        return response.json();
      })
      .then(function(resp) {
        document.getElementById("Output").innerHTML =
JSON.stringify(resp.data);
      })
      .catch(function(resp) {
      document.getElementById("Output").innerHTML =
"There was an error";
      });
}
```

EXAMPLES

```
// building a placeholder example
let inp = form.elements.searchBox;

inp.value = "Enter your search term here";
inp.focus();

inp.addEventListener('focus', function(){
  if (inp.value==="Enter your search term here") {
    inp.value = "";
  }
});

inp.addEventListener('blur', function(){
  if(inp.value == "") {
    inp.value = "Enter your search term here";
    }
});
```

```
// get all values of a dropdown
const input_type = form.elements.searchType;
const form_type_vals = [];
for (let i=0; i < input_type.length; i++) {
  if (input_type[i].selected) {
    form_type_vals.push(input_type[i].value);
}};
```

EXAMPLES

```
// validation example…

form.addEventListener("submit",validate);
function validate(e) {
  var inp = form.elements.searchBox.value;
  if (inp === "Enter your search term here") {
      e.preventDefault();
      alert("Your search term cannot be empty!");
  }
}
```

EXAMPLES

```
// one that works with any field marked as required-field (remember to mark them in html)

form.addEventListener("submit",validate);
function validate(e) {
  let form = e.target,
      error = "";

  for (let i = 0, el = form.elements.length; i < el; i++) {
    let element = form.elements[i];
    console.log(element.type, element.value, element.checked,
element.hasAttribute("required-field"));

    if (element.hasAttribute("required-field")) {
      if (element.type === "text" && (element.value === "" || element.value === "Enter
your search term here")) {
        error += "Your search term cannot be empty! ";
      }

      if (element.type === "password" && element.value === "" ) {
        error += "Your secret word cannot be empty! ";
      }

      if (element.type === "select-one" && element.value === "" ) {
        error += "Your search type cannot be empty! ";
      }
    }
  }

  // for checkboxes and other multiple inputs, this is not ideal. Do it throgh
queryselectors
  if (document.querySelectorAll('input[required-field]').length > 0) {
    let check_checked =
document.querySelectorAll('input[type=checkbox][required-field]:checked');

    if (check_checked.length === 0 ) {
      error += "Your search area cannot be empty! ";
    }
  }

  console.log(error);
  if (error !== "") {
    alert(error);
    e.preventDefault();
  }
}
```

EXAMPLES

```
//try with inline validation excercise
inp.addEventListener("blur",inlineValidate);
function inlineValidate(e) {
  let inp_field = e.target;
  if (inp_field.value === "Enter your search term here"
|| inp_field.value === "") {
      let error_label = document.createElement('p');
      error_label.innerHTML = "Your search term cannot be
empty";
      error_label.classList.add('error');
      if
(inp_field.parentNode.getElementsByClassName('error').len
gth > 0) {

console.log(inp_field.parentNode.getElementsByClassName('
error')[0]);
        inp_field.parentNode.replaceChild(error_label,
inp_field.parentNode.getElementsByClassName('error')[0]);

      } else {
        inp_field.parentNode.insertBefore(error_label,
inp_field.nextSibling);
      }
  } else {
      if
(inp_field.parentNode.getElementsByClassName('error').len
gth > 0) {

inp_field.parentNode.removeChild(inp_field.parentNode.get
ElementsByClassName('error')[0]);
      }
  }
}
```

EXAMPLES

```
//ajax testing with https://reqres.in/
let button = document.getElementById('GetUsers');
button.addEventListener("click", getUserData);

function getUserData() {
  let url = "https://reqres.in/api/users";
  let xhr = new XMLHttpRequest();

  xhr.onreadystatechange = function() {

    if (xhr.readyState === 0 ) {
      console.log('request created');
      document.getElementById("Output").innerHTML = "request created";
    }

    if (xhr.readyState === 1 ) {
      console.log('request sending');
      document.getElementById("Output").innerHTML = "request sending";
    }

    if (xhr.readyState === 2 ) {
      console.log('headers received');
      document.getElementById("Output").innerHTML = "headers received";
    }

    if (xhr.readyState === 3 ) {
      console.log('response received');
      document.getElementById("Output").innerHTML = "response received";
    }

    if (xhr.readyState === 4 ) {
      if (xhr.status === 200) {
        document.getElementById("Output").innerHTML = xhr.responseText;
      } else {
        document.getElementById("Output").innerHTML = "There was an error";
      }
    }
  }

  xhr.open("GET", url, true);
  xhr.send(null);
  console.log(xhr);
}
```

EXAMPLES

```
const form = document.getElementById('createUser')

form.addEventListener("submit", saveUserData);

function saveUserData(e) {
  e.preventDefault();

  let url = "https://reqres.in/api/users";

  let xhr = new XMLHttpRequest();
  let user = {}; // create an empty object
  user.name = form.first_name.value + ' ' + form.last_name.value;
  console.log(user, JSON.stringify(user));
  xhr.onreadystatechange = function() {
    if (xhr.readyState === 4 ) {
      if (xhr.status === 200 || xhr.status === 201) {
        let resp = JSON.parse(xhr.response);
        document.getElementById("Output").innerHTML = "Successfully
created id: "+resp.id;
      } else {
        document.getElementById("Output").innerHTML = "There was an
error";
      }
    }
  }

  xhr.open("POST", url, true);
  xhr.setRequestHeader("Content-Type", "application/json");
  xhr.send(JSON.stringify(user));
  console.log(xhr);
}
```

EXAMPLES

```
// exercise: use get above to get data from reqres and then format
in the DOM. based on example from class 1

var ul = document.createElement('ul');
var url = 'https://randomuser.me/api/?results=10';
var xhr = new XMLHttpRequest();
xhr.onreadystatechange = function() {
    if (xhr.readyState == XMLHttpRequest.DONE) {
        var authors = JSON.parse(xhr.responseText); // Getresults
        for (key in authors.results) { // loop through theresults
          var author = authors.results[key]; //assign current row
to author var
          var li = document.createElement('li'), //  Create
theelements we need
              img = document.createElement('img'),
              span = document.createElement('span');
          img.src = author.picture.medium;  // Add the source ofthe
image to be the src of the img element
          span.innerHTML = author.name.first + ' '
+author.name.last; // Make the HTML of our span to be the firstand
last name of our author
          li.appendChild(img); // Append img element back
tocontaining li
          li.appendChild(span); // Append span element back
tocontaining li
          ul.appendChild(li); // Append li element back
tocontaining ul
          document.body.append(ul); //Append the new ul to body
} }
}
xhr.open('GET', url, true);
xhr.send(null);
```

EXAMPLES

```
// fetch get
button.addEventListener("click", getUserData);

function getUserData() {
  let url = "https://reqres.in/api/users";
  fetch(url)
    .then(function(response) {
      return response.json();
    })
    .then(function(resp) {
      document.getElementById("Output").innerHTML =
JSON.stringify(resp.data);
    })
    .catch(function(error) {
      document.getElementById("Output").innerHTML = "There was an
error "+error;
    });
}
```

EXAMPLES

```
// fetch post
form.addEventListener("submit", saveUserData);

function saveUserData(e) {
  e.preventDefault();

  const url = "https://reqres.in/api/users";
  const FD  = new FormData(form);
  FD.append("name",form.first_name.value + ' ' +
form.last_name.value);
  let jsonObject = {};
  for (let pair  of FD.entries()) {
      jsonObject[pair[0]] = pair[1];
  }
  //const req = ;
  console.log(jsonObject);
  //console.log(req);
  fetch(url, {
    method: 'POST',
    headers: {'Content-Type': 'application/json'},
    body: JSON.stringify(jsonObject)
  })
    .then(function(response) {
        //console.log(response.json());
        return response.json();
    })
    .then(function(data) {
        console.log('raw data',data);
        document.getElementById("Output").innerHTML = "Successfully
created id: "+data.id;
    })
    .catch(function(error) {
        document.getElementById("Output").innerHTML = "Th1ere was
an error "+error;
    });
}
```

Assignment

To test out AJAX, build a simple form where you can enter in any keywords or phrases. Then perform a button click or submit action to capture that phrase and use ajax to send that data to the Google Custom Search API (or any other API of your choice that returns dta that can be tabulated, formatted and generally acted upon). Format the returned data (probably would be in JSON format) and present the first result (or more) below the asked question. Format the markup as needed.

Next Steps

➔ OOP Concepts