03. JSON i LOCALSTORAGE

IES L'ESTACIÓ 2021-2022

JSON i LOCALSTORAGE.

- 1. ¿Què és JSON?.
  - 1.1 Intercanvi de dades.
  - 1.2 Enviant dades.
  - 1.3 Rebent dades.
  - 1.4 Emmagatzematge de dades.
- 2. Sintaxi JSON.
  - 2.1 DadeS JSON: un nom i un valor.
  - 2.2. JSON s'avalua en objectes JS.
  - 2.3 Valors JSON.
  - 2.4. JSON utiliza la sintaxis de JavaScript
- 3. Tipus de dades JSON.
- 4. JSON.parse()
- 5. JSON.stringify()
- 6. Objectes JSON.
- 7. Arrays JSON.
- 8. LocalStorage.

1. ¿Què és JSON?.

- **JSON** (Java Script Object Notation) és un format lleuger d'intercanvi de dades.
- Llegir-ho i escriure-ho és simple per a humans, mentre que per a les màquines és simple interpretar-ho i generar-ho.
- Està basat en un subconjunt del Llenguatge de Programació Javascript.
- **JSON** és un format de text que és completament independent del llenguatge, però utilitza convencions que són àmpliament coneguts pels programadors de la família de llenguatges C, incloent-hi C, C++, C#, Java, Javascript, Perl, Python, i molts altres.
- El tipus d'arxiu dels arxius **JSON** és ".json“
- El tipus MIME per al text **JSON** és "application/json"

1. ¿Què és JSON?.

- **1.1 Intercanvi de dades.**

- En intercanviar dades entre un navegador i un servidor, les dades només poden ser text.

- **JSON** és text i podem convertir qualsevol objecte Javascript en **JSON** i enviar **JSON** al servidor.

- També podem convertir qualsevol **JSON** rebut del servidor en objectes Javascript.

- D'aquesta manera podem treballar amb les dades com a objectes Javascript, sense complicades anàlisis i traduccions.

1. ¿Què és JSON?.

- **1.2 Enviant dades.**

- Si té dades emmagatzemades en un objecte Javascript, pot convertir l'objecte en **JSON** i enviar-lo a un servidor:

Exemple

```
var myObj = {name: "John", age: 31, city: "New York"};
var myJSON = JSON.stringify(myObj);
window.location = "demo_json.php?x=" + myJSON;
```

1. ¿Què és JSON?.

- **1.3 Rebent dades.**

- Si rep dades en format JSON, pot convertir-se en un objecte JavaScript:

Exemple

```
var myJSON = '{"name":"John", "age":31, "city":"New York"}';
var myObj = JSON.parse(myJSON);
document.getElementById("demo").innerHTML = myObj.name;
```

1. ¿Què és JSON?.

- **1.4 Emmagatzematge de dades.**

- A l'emmagatzematge de dades, les dades han de tenir un format determinat i, independentment d'on es troba emmagatzemat, el text sempre és un dels formats legals.

- JSON permet emmagatzemar objectes JavaScript com a text.

Exemple

Almacenar dades en emmagatzematge local

```
// Storing data:
myObj = {name: "John", age: 31, city: "New York"};
myJSON = JSON.stringify(myObj);
localStorage.setItem("testJSON", myJSON);

// Retrieving data:
text = localStorage.getItem("testJSON");
obj = JSON.parse(text);
document.getElementById("demo").innerHTML = obj.name;
```

2. Sintaxi JSON.

- La sintaxi **JSON** es deriva de la sintaxi de notació d'objectes de Javascript:
  - Les dades estan en parells de nom / valor.
  - Les dades estan separades per comes.
  - {} objectes.
  - [] arrays.
  - **2.1. Dades JSON: un nom i un valor.**
    - Les dades **JSON** s'escriuen com a parells de nom / valor.
    - Un parell de nom / valor consta d'un nom de camp (entre cometes dobles), seguit de dos punts, seguit d'un valor:
    - **Exemple:**

      “name” : ”John”

Nota: Els noms JSON requereixen cometes dobles. Els noms de Javascript no ho fan.

2. Sintaxi JSON.

- **2.2. JSON s'avalua en objectes JS.**
  - El format JSON és quasi idèntic als objectes Javascript.
  - En JSON, les claus han de ser cadenes, escrites amb cometes dobles:
  - **Exemple JSON:**

    { "name" : "John" }

  - En JavaScript, les claus poden ser cadenes, números o noms d'identificadors:
  - **Exemple Javascript:**

    { name : "John" } o { name : 'John' }

2. Sintaxi JSON.

- **2.3. Valors JSON**
  - En JSON els valors han de ser un dels següents tipus de dades:
    - String.
    - Number.
    - Object
    - Array.
    - Boolen
    - Null
  - En Javascript, els valors poden ser tots els anteriors, més qualsevol altra expressió Javascript vàlida, que inclou:
    - Function.
    - Date.
    - Undefined

2. Sintaxi JSON.

- **2.4. JSON utiliza la sintaxis de JavaScript**
  - Pel fet que la sintaxi JSON es deriva de la notació d'objectes de Javascript, es necessita molt poc software addicional per a treballar amb JSON dins de Javascript.
  - Amb Javascript es pot crear un objecte i assignar-li dades, així:
    - **var person = { name: "John", age: 31, city: "New York" };**
  - Pot acceder a un objecte JavaScript com aquest:
    - **person.name**; *//returns John*
  - També es pot accedir així:
    - **person[“name”]**; *//returns John*
  - Les dades es poden modificar així:
    - **person.name= “Gilbert”**;
  - També es pot modificar així:
    - **person[“name”]=“Gilbert”**;

3. Tipus de dades JSON.

- Strings: han d'escriure's entre cometes dobles**. Ex: { "name":"John" }**
- Numbers: han de ser int o float. **Ex: { "age":30 }**
- Booleans: true or false **Ex: { "sale":true }**
- Null: **Ex: { "middlename":null }**
- Objectes: **Ex: {**

  "employee":{ "name":"John", "age":30, "city":"New York" }

  }

- Matrius: **Ex: {**

  "employees":["John", "Anna", "Peter"]

  }

4. JSON.parse()

- Quan reben dades d'un servidor web, les dades són un string.
- Utilitzar la funció **JSON.parse()** consisteix a passar el string rebut del servidor a un objecte Javascript.
- Exemple:
  - Text rebut del servidor:
    - **'{ "name":"John", "age":30, "city":"New York"}'** .
  - Utilitzen la funció JavaScript JSON.parse() per convertir text en un objecte JavaScript:
    - **var obj = JSON.parse('{ "name":"John", "age":30, "city":"New York"}');**

4. JSON.parse()

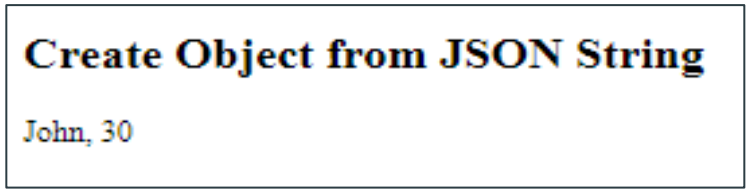
```
<!DOCTYPE html>
<html>
<body>

<h2>Create Object from JSON String</h2>

<p id="demo"></p>

<script>
var txt = '{"name":"John", "age":30, "city":"New York"}'
var obj = JSON.parse(txt);
document.getElementById("demo").innerHTML = obj.name + ", " + obj.age;
</script>

</body>
</html>
```



4. JSON.parse()

- **Excepcions:**
- Els objectes data no estan permesos en JSON, s'ha d'escriure com un String i convertir-ho en un objecte data més tard.

```
<!DOCTYPE html>
<html>
<body>

<h2>Convert a string into a date object.</h2>

<p id="demo"></p>

<script>
var text = '{"name":"John", "birth":"1986-12-14", "city":"New York"}';
var obj = JSON.parse(text);
obj.birth = new Date(obj.birth);
document.getElementById("demo").innerHTML = obj.name + ", " + obj.birth;
</script>

</body>
</html>
```

4. JSON.parse()

- **Excepcions:**
- No es permeten funcions en JSON, si s'ha d'incloure una funció, s'haurà d'escriure com un String i per a tornar a convertir-ho més tard.

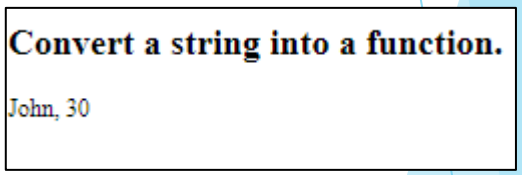
```
<!DOCTYPE html>
<html>
<body>

<h2>Convert a string into a function.</h2>

<p id="demo"></p>

<script>
var text = '{"name":"John", "age":"function() {return 30;}", "city":"New York"}';
var obj = JSON.parse(text);
obj.age = eval("(" + obj.age + ")");
document.getElementById("demo").innerHTML = obj.name + ", " + obj.age();
</script>

</body>
</html>
```



5. JSON.stringify()

- Un ús comú de JSON és intercanviar dades cap a / des d'un servidor web.
- En enviar dades a un servidor web, les dades han de ser un String.
- Convertim un objecte Javascript en un String amb **JSON.stringify().**
- Exemple:
- Tenim un objecte Javascript:
  - **var obj = JSON.parse('{ "name":"John", "age":30, "city":"New York"}');**
- Utilitzen la funció JavaScript JSON.stringify() per convertir un objecte JavaScript en text:
  - **var myJSON = JSON.stringify(obj);**

5. JSON.stringify()

```
<!DOCTYPE html>
<html>
<body>

<h2>Create JSON string from a JavaScript object.</h2>

<p id="demo"></p>

<script>
var obj = { name: "John", age: 30, city: "New York" };
var myJSON = JSON.stringify(obj);
document.getElementById("demo").innerHTML = myJSON;
</script>

</body>
</html>
```

Create JSON string from a JavaScript object.

{"name":"John","age":30,"city":"New York"}

5. JSON.stringify()

- També és possible seqüenciar matrius de Javascript:
- Exemple:
- Tenim aquesta matriu en Javascript:
  - **var arr = ["John", "Peter", "Sally", "Jane"];**
- Utilitzen la funció JavaScript JSON.stringify() per convertir un objecte JavaScript en text:
  - **var myJSON = JSON.stringify(arr);**

5. JSON.stringify()

```
<!DOCTYPE html>
<html>
<body>

<h2>Create JSON string from a JavaScript array.</h2>

<p id="demo"></p>

<script>

var arr = [ "John", "Peter", "Sally", "Jane" ];
var myJSON = JSON.stringify(arr);
document.getElementById("demo").innerHTML = myJSON;

</script>

</body>
</html>
```

Create JSON string from a JavaScript array.

["John","Peter","Sally","Jane"]

4. JSON. stringify()

- **Excepcions:**
- Els objectes data no estan permesos en JSON, la funció **JSON.stringify()** convertirà qualsevol data en cadenes.

```
<!DOCTYPE html>
<html>
<body>

<h2>JSON.stringify will convert any date objects into strings.</h2>

<p id="demo"></p>

<script>
var obj = { name: "John", today: new Date(), city: "New York" };
var myJSON = JSON.stringify(obj);
document.getElementById("demo").innerHTML = myJSON;
</script>

</body>
</html>
```

JSON.stringify will convert any date objects into strings.

{"name":"John","today":"2021-05-19T10:45:17.270Z","city":"New York"}

4. JSON.stringify()

- **Excepcions:**
- No es permeten funcions com a valors d'objecte en JSON, la funció JSON.stringify() eliminarà qualsevol funció d'objecte Javascript, tant la clau com el valor:

```
<!DOCTYPE html>
<html>
<body>

<h2>JSON.stringify will remove any functions from an object.</h2>

<p id="demo"></p>

<script>
var obj = { name: "John", age: function () {return 30;}, city: "New York" };
var myJSON = JSON.stringify(obj);
document.getElementById("demo").innerHTML = myJSON;
</script>

</body>
</html>
```

JSON.stringify will remove any functions from an object.

{"name":"John","city":"New York"}

4. JSON.stringify()

- **Excepcions:**
- Això es pot ometre si converteix les seues funcions en cadenes abans d'executar la funció JSON.stringify():

```
<!DOCTYPE html>
<html>
<body>

<h2>JSON.stringify will remove any functions from an object.</h2>

<p>Convert the functions into strings to keep them in the JSON object.</p>

<p id="demo"></p>

<script>
var obj = { name: "John", age: function () {return 30;}, city: "New York" };
obj.age = obj.age.toString();
var myJSON = JSON.stringify(obj);
document.getElementById("demo").innerHTML = myJSON;
</script>

</body>
</html>
```

JSON.stringify will remove any functions from an object.

Convert the functions into strings to keep them in the JSON object.

{"name":"John","age":"function () {return 30;}","city":"New York"}

6. Objectes JSON

- **6.1 Sintaxi de l'objecte.**
  - Els objectes JSON estan envoltats per claus {}.
  - Els objectes JSON s'escriuen en parells clau / valor.
  - Les claus han de ser cadenes i els valors han de ser un tipus de dades JSON vàlid (String, number, objecte, matriu, booleà o nul).
  - Les claus i els valors estan separats per dos punts.
  - Cada parell clau / valor està separat per una coma.
  - Exemple:

```
{ "name":"John", "age":30, "car":null }
```

6. Objectes JSON

- **6.2 Accedir als valors dels objectes.**
  - Pot accedir als valors de l'objecte mitjançant la notació de punts (.):

```
myObj = { "name":"John", "age":30, "car":null };
x = myObj.name;
```

  - També pot accedir als valors de l'objecte utilitzant la notació entre claudàtors ([]):

```
myObj = { "name":"John", "age":30, "car":null };
x = myObj["name"];
```

6. Objectes JSON

- **6.3 Bucle d'un objecte.**
  - Pot recórrer les propietats de l'objecte utilitzant el bucle for-in:

```
myObj = { "name":"John", "age":30, "car":null };
for (x in myObj) {
  document.getElementById("demo").innerHTML += x;
}
```

  - En un bucle for-in, use la notació de claudàtors per a accedir als valors de propietat :

```
myObj = { "name":"John", "age":30, "car":null };
for (x in myObj) {
  document.getElementById("demo").innerHTML += myObj[x];
}
```

6. Objectes JSON

- **6.4 Objectes JSON anidats.**
  - Els valors d'un objecte JSON poden ser un altre objecte JSON.

```
myObj = {
  "name":"John",
  "age":30,
  "cars": {
    "car1":"Ford",
    "car2":"BMW",
    "car3":"Fiat"
  }
 }
```

  - Pot accedir als objectes JSON anidats mitjançant la notació de punts o la notació de claudàtors:

```
x = myObj.cars.car2;
// or:
x = myObj.cars["car2"];
```

6. Objectes JSON

- **6.5 Modificar valors.**
  - Pot usar la notació de punts per a modificar qualsevol valor en un objecte JSON:

```
myObj.cars.car2 = "Mercedes";
```

  - També pot usar la notació entre claudàtors per a modificar un valor en un objecte JSON:

```
myObj.cars["car2"] = "Mercedes";
```

- **6.6 Eliminar valors.**
  - Utilitzent la paraula clau **delete** per a eliminar propietats d'un objecte JSON:

```
delete myObj.cars.car2;
```

7. Arrays JSON

- **7.1 Arrays com objectes JSON.**
  - Els Arrays en JSON són pràcticament idèntics als de Javascript.
  - En JSON, els valors del Array han de ser de tipus String, number, object, array, booleà o null.
  - En Javascript, els valors del array poden ser tots els anteriors, més qualsevol altra expressió de Javascript vàlida, incloses funcions, dates i indefinides.
  - Exemple:

```
[ "Ford", "BMW", "Fiat" ]
```

7. Arrays JSON

- **7.2 Arrays en objectes JSON.**
  - Els Arrays podem ser valors d'una propietat de l'objecte.
  - Exemple:

```
{
"name":"John",
"age":30,
"cars":[ "Ford", "BMW", "Fiat" ]
}
```

- **7.3 Accedir als valors de l'Array.**
  - Podem accedir als valors de l'Array utilitzant el número d'índex:
  - Exemple:

```
x = myObj.cars[0];
```

7. Arrays JSON

- **7.4 Bucles en Arrays.**
  - Podem accedir als valors de l'Array utilitzant un bucle **for-in** :
  - Exemple:

```
<!DOCTYPE html>
<html>
<body>

<p>Looping through an array using a for in loop:</p>

<p id="demo"></p>

<script>
var myObj, i, x = "";
myObj = {
  "name":"John",
  "age":30,
  "cars":[ "Ford", "BMW", "Fiat" ]
};

for (i in myObj.cars) {
  x += myObj.cars[i] + "<br>";
}
document.getElementById("demo").innerHTML = x;
</script>

</body>
</html>
```

Looping through an array using a for in loop:

Ford
BMW
Fiat

7. Arrays JSON

- **7.4 Bucles en Arrays.**
  - O podem usar un bucle **for**:
  - Exemple:

```
<!DOCTYPE html>
<html>
<body>

<p>Loopin through an array using a for loop:</p>

<p id="demo"></p>

<script>
var myObj, i, x = "";
myObj = {
  "name":"John",
  "age":30,
  "cars":[ "Ford", "BMW", "Fiat" ]
};

for (i = 0; i < myObj.cars.length; i++) {
  x += myObj.cars[i] + "<br>";
}
document.getElementById("demo").innerHTML = x;
</script>

</body>
</html>
```

Loopin through an array using a for loop:

Ford
BMW
Fiat

7. Arrays JSON

- **7.5 Arrays anidats en objectes JSON.**
  - Els valors d'un Array també poden ser un altre Array o fins i tot un altre objecte JSON:
  - Exemple:

```
myObj = {
  "name":"John",
  "age":30,
  "cars": [
    { "name":"Ford", "models":[ "Fiesta", "Focus", "Mustang" ] },
    { "name":"BMW", "models":[ "320", "X3", "X5" ] },
    { "name":"Fiat", "models":[ "500", "Panda" ] }
  ]
 }
```

7. Arrays JSON

- **7.5 Arrays anidats en objectes JSON.**
  - Per a accedir als Arrays dins d'aquets, utilitzen un bucle for-in per a cada Array:
  - Exemple:

```
<!DOCTYPE html>
<html>
<body>

<p>Looping through arrays inside arrays.</p>

<p id="demo"></p>

<script>
var myObj, i, j, x = "";
myObj = {
  "name":"John",
  "age":30,
  "cars": [
    {"name":"Ford", "models":["Fiesta", "Focus", "Mustang"]},
    {"name":"BMW", "models":["320", "X3", "X5"]},
    {"name":"Fiat", "models":["500", "Panda"] }
  ]
}
for (i in myObj.cars) {
  x += "<h2>" + myObj.cars[i].name + "</h2>";
  for (j in myObj.cars[i].models) {
    x += myObj.cars[i].models[j] + "<br>";
  }
}
document.getElementById("demo").innerHTML = x;
</script>

</body>
</html>
```

Looping through arrays inside arrays.

Ford

Fiesta
Focus
Mustang

BMW

320
X3
X5

Fiat

500
Panda

7. Arrays JSON

- **7.6 Modificar valors d'un Array.**
  - Utilitze el número d'índex per a modificar un Array:
  - Exemple:

```
myObj.cars[1] = "Mercedes";
```

- **7.7 Eliminar valors d'un Array.**
  - S'utilitza la paraula clau **delete** per a eliminar elements d'un Array:
  - Exemple:

```
delete myObj.cars[1];
```

8. ¿Què és LocalStorage?

- Utilitzant **LocalStorage** podem guardar informació en el nostre navegador web a mode de sessió i que eixa informació persistisca i estiga disponible durant la navegació entre les diferents pàgines del nostre lloc o aplicació web.

- El **LocalStorage** sol usar-se molt en aplicacions web desenvolupades completament amb Javascript, amb tecnologies com a **Angular**, encara que també pot aplicar-se a qualsevol web en la qual necessitem compartir dades entre seccions.

- En les aplicacions web monolítiques desenvolupades amb un llenguatge de backend com PHP o qualsevol altre, l'ús del LocalStorage és sustituit per les sessions del propi llenguatge de backend ja que el frontend i el backend estan completament integrats i barrejats.

- Vegem com usar el LocalStorage en Javascript.

8. ¿Què és LocalStorage?

- **Comprovar si el navegador és compatible:**
  - Podem veure si el navegador web té disponible la funcionalitat del localstorage així:

```
if (typeof(Storage) !== "undefined") {
    // LocalStorage avalaible
} else {
    // LocalStorage not supported in this browser
}
```

- **Guardar dades en el navegador:**
  - Per a emmagatzemar dades i guardar nous elements o indexes en el LocalStorage usarem la següent instrucció:

```
// Save
localStorage.setItem("title", "Curs JavaScript");
```

  - D'aquesta manera donem d'alta un nou element en l'emmagatzematge del browser.

8. ¿Què és LocalStorage?

- **Recuperar dades**
  - Per a aconseguir les dades que tenim guardats en un indexe del nostre emmagatzematge local del navegador usarem:

```
// Get element
localStorage.getItem("title");
```

  - Això ens retornarà el valor que hem guardat anteriorment.
- **Guardar objectes en el LocalStorage**
  - Per a guardar un objecte primer hem de convertir-lo en un string json ja que el localstorage no permet guardar objectes de JavaSciprt com a tal.
  - Hauríem de fer una cosa així:

```
localStorage.setItem("user", JSON.stringify(my_object));
```

  - Guardem l'element usuari el valor del qual és un objecte convertit a string.

8. ¿Què és LocalStorage?

- **Traure objectes del LocalStorage**
  - Per a recuperar un objecte primer hem de convertir-lo en un objecte de Javascript json en lloc del string json que hi ha guardat per defecte.

```
JSON.parse(localStorage.getItem("user"));
```

  - Recordem que:
    - JSON.parse() és per a analitzar o convertir alguna cosa a un objecte JSON usable per Javascript.
    - JSON.stringify() és per a crear un JSON string d'un objecte o un array**.**
- **Borrar o buidar el localStorage**
  - Per a eliminar un element del localStorage farem:

```
localStorage.removeItem("title");
```

  - Per a eliminar totes les variables guardades en el *localStorage farem:

```
localStorage.clear();
```