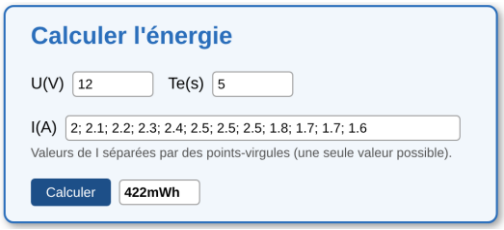
Technologies du Web

Tutoriel « Découverte du langage JavaScript »

CONNAISSANCES ABORDÉES

- Déclaration, initialisation et utilisation de variables.
- Structures alternatives, fonction, évènements.
- Structure de données : tableau.
- Entrées / sorties d'information avec des boîtes de dialogue (prompt, alert) ou un formulaire.



OBJECTIF

Calculer l'énergie électrique consommée par un circuit alimenté en courant continu et l'afficher dans une IHM.

CONDITIONS DE DÉROULEMENT DE L'ACTIVITÉ

Étapes	*Objectifs pédagogiques*	*Activités pratiques*
1 - Mise en situation	**Comprendre** le principe de calcul de l'énergie électrique en courant continu.	Lecture de la mise en situation.
2 - Automatiser le calcul de l'énergie électrique	**Calculer** et **afficher** la valeur de l'énergie consommée par un circuit électrique alimenté en courant continu. - La **tension** et **l'intensité** sont **constantes**. - La **tension** est **constante.** L'**intensité varie** dans le **temps**. **Rédiger** une documentation.	**Codage en JavaScript** - Déclaration, initialisation et utilisation de variables. - Programmation d'une structure alternative. - Écriture d'une fonction. - Création d'un formulaire et association d'un évènement à un bouton. - Calcul d'une énergie électrique. Changement d'unités. **Codage en Markdown** Rédaction des algorithmes en pseudo-code.
3 - Synthèse	**Calculer** l'énergie électrique consommée par un circuit alimenté en courant continu et l'afficher dans une IHM.	**Codage en JavaScript**.

SIGLES UTILISÉS

: Information : Éditeur de code (voir prof)

PRÉPARATION DU DOSSIER DE TRAVAIL

Organisez votre dossier de travail en suivant les indications données dans l'**annexe 3** située à la fin de ce document.

LOGICIELS ET MATÉRIELS

PC, navigateur (Edge, Chrome, Opera, **Firefox**, etc.), éditeur de code.

Table des matières

1 Mise en situation 3

1.1 Rappels 3

1.2 Comment calculer l'énergie électrique dans un circuit alimenté en courant continu ? 3

1.3 Travail réalisé dans le TP 3

2 Calculer l'énergie électrique 4

2.1 Présentation du langage JavaScript 4

2.2 Programme version 1 : la tension et l'intensité sont constantes 5

2.3 Programme version 2 : changement des unités de E 6

2.4 Programme version 3 : le programme précédent devient une fonction 6

2.5 Programme version 4 : formulaire et évènements 7

2.6 Programme version 5 : la tension est constante, l'intensité varie dans le temps 8

3 Synthèse 9

Annexe 1 - Résumé JavaScript 10

Annexe 2 – Exemple de traitement d'un formulaire en JavaScript 11

Annexe 3 - Installation du dossier du TP dans votre compte sur le serveur Nas_SIN 11

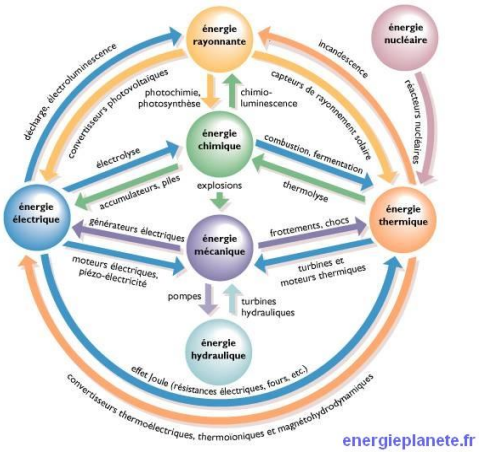
RESSOURCES

- Guide JavaScript sur MDN — MDN, guide JavaScript
- Markdown sur Wikipédia — Wikipédia, Markdown

---

https://webge.fr/web6a/tuto

1 Mise en situation



On souhaite mesurer la consommation électrique d'un moteur à courant continu alimenté en 12 V pendant une durée donnée.

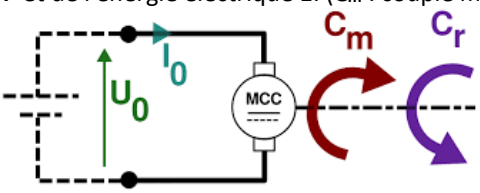
À consulter avant de poursuivre : https://webge.fr/web6a

1.1 Rappels

- L'**énergie**, comme le **travail** qu'elle peut produire, se mesure en **joules** (J). Elle se note souvent **E**.
  Exemple : E = 450 J
  Dans certains cas, on utilise d'autres unités comme le wattheure :

1 Wh = 3600 J

- **L'énergie électrique dans un circuit alimenté en courant continu**
  Si un récepteur est soumis à une tension U_0 et qu'il est traversé par un courant d'intensité I_0 pendant un temps **t** alors il va absorber une puissance **P** et de l'énergie électrique **E**. (C_m : couple moteur, C_r : couple résistant)

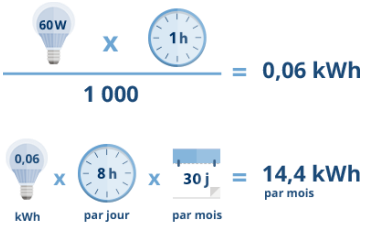


En **courant continu** :

$$P_{(W)}=U_{(V)} \times I_{(A)}$$

$$E_{(J)}=U_{(V)} \times I_{(A)} \times t_{(s)}$$

Avec : U en volts, I en ampères, t en secondes, P en watts et E en joules.



1.2 Comment calculer l'énergie électrique dans un circuit alimenté en courant continu ?

En courant continu ou pour un appareil résistif, l'énergie consommée peut être obtenue en **mesurant la tension** et **l'intensité du courant** à l'aide de **capteurs** et le **temps** avec un **timer** ou une **horloge temps réel.** Le calcul est réalisé par un **logiciel de traitement**.



Ce logiciel peut être embarqué dans un microcontrôleur pour un affichage local (au plus près du circuit électrique) ou un affichage distant, par exemple dans un **site web**. Les informations **U, I et t** sont alors **transmises à distance**.

1.3 Travail réalisé dans le TP

Vous allez **concevoir**, en **JavaScript**, le traitement à appliquer à la mesure des grandeurs U, I et t pour afficher l'énergie E dans un navigateur.

Vous allez également **documenter** vos codes en éditant des fichiers **Markdown**.

2 Calculer l'énergie électrique

2.1 Présentation du langage JavaScript



*« JavaScript (JS) est un langage de programmation principalement utilisé côté client pour générer des **pages web dynamiquement**, mais également côté serveur, depuis l'arrivée de Node.js.*
Ne pas confondre JavaScript avec le langage de programmation Java car les deux langages de programmation ont une syntaxe, une sémantique et des utilisations très différentes.



*Le JavaScript a été inventé par **Brendan Eich**, co-fondateur du projet Mozilla, de la Mozilla Foundation et de la Mozilla Corporation. »*

JavaScript est l'un des langages les plus utilisés au monde, grâce au développement et à l'amélioration des performances des API dans les navigateurs.

Où placer le code JavaScript ?

MDN, ajouter du JavaScript

- **Cas 1. Le code JavaScript dans la page web**
  On place le code JavaScript au sein de l'élément **<script>** comme ci-dessous. Cet élément peut être placé à l'extérieur de body.



```
<!DOCTYPE html>
<html>
<head>
    <title>Hello World!</title>
</head>
<body>
    <script>
        alert('Hello world!');
    </script>
</body>
</html>
```

Exemple

- **Cas 2. Le code JavaScript dans un script externe**
  Le script est placé dans un fichier avec l'extension .js. Un lien vers ce fichier est placé dans le fichier HTML comme ci-dessous.

HTML (index.html)

```
<!DOCTYPE html>
<html>
<head>
    <title>Hello World!</title>
</head>
<body>
    <script src="chemin/vers/le/script.js"></script>
</body>
</html>
```

Fichier JavaScript externe (script.js)

```
alert('Hello world!');
```

Exemple

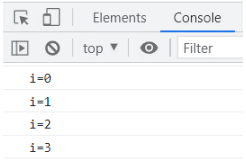
Comment tester le code dans le navigateur ?

MDN, console

« En informatique, l'**instrumentation** du ***code source*** est une opération consistant à ajouter des instructions-machine supplémentaires à un programme dans le but de le tester. »



En JavaScript l'objet **console** donne accès à la console de débogage du navigateur. On peut accéder à celle de Chrome avec la touche **F12**.



Exemple : affichage du contenu d'une variable dans la console de Chrome.

```
for (let i = 0; i < 10; i++) {
  console.log("i = " + i);
}
```

2.2 Programme version 1 : la tension et l'intensité sont constantes

Objectifs : être capable de déclarer des variables et d'utiliser les boîtes de dialogue prompt/alert.

Spécification

Nom : EnergiejsV1
Rôle : calculer et afficher l'énergie consommée par un circuit alimenté en courant continu, avec U et I constants.
Données d'entrée : U, tension en V, réel ≥ 0 ; I, intensité en A, réel ≥ 0 ; t, durée en s, entier ≥ 0
Donnée de sortie : E, réel, énergie consommée en J
Traitement : E ← U × I × t : en J
Préconditions : U, I et t sont des valeurs numériques positives ou nulles.
Postconditions : E = U × I × t et E ≥ 0.

```
Algorithme
EnergiejsV1
Variables
E, I, U : réels
t : entier

début
    lire(U,I,t)
    E ← U * I * t
    écrire(E)
fin
```

A. Codage en JavaScript

Implémentez l'algorithme **EnergiejsV1** dans le fichier **index.html** du répertoire **v1** en suivant la démarche ci-dessous.

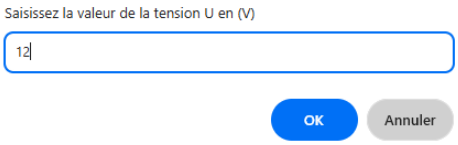
- Consultez l'**annexe 1, §1.** Créez les quatre variables U, I, t et E.
  Pour en savoir plus : MDN, déclarations
- Consultez l'**annexe 1, §2.** *Écrivez le code de saisie* des valeurs U, I, t avec des boîtes de dialogue **prompt.**
  Pour en savoir plus : MDN, prompt()

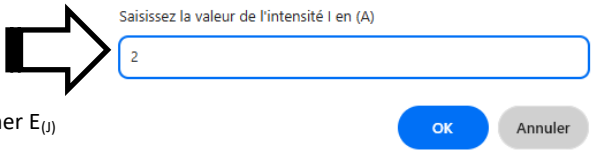
⚠ prompt() renvoie du texte. Convertissez-le avec **parseFloat() ou parseInt()** selon le type de la variable (**annexe 1, §2**).

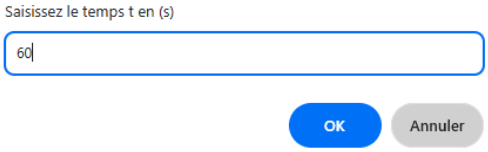
- Écrivez le code de calcul de l'énergie $E_{(J)}$.
- Consultez la ressource ci-dessous et écrivez le code pour afficher $E_{(J)}$ dans une boîte de dialogue **alert.**
  ***Ressource** : **annexe 1, §2** (en savoir plus : MDN, alert())*

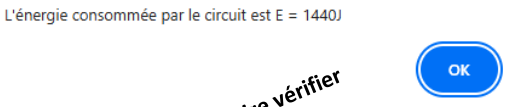
Remarque : voir également la **concaténation de chaînes** pour attacher le résultat du calcul sous la forme : *E = 1440J*.
***Ressource** : **annexe 1, §2** (en savoir plus : MDN, chaînes)*

Exemple de résultat attendu (jeu d'essai)









À faire vérifier

B. Documentation en Markdown

*"**Markdown** est un **langage de balisage léger** créé en 2004 par John Gruber avec l'aide d'Aaron Swartz.*

*Son but est d'offrir une syntaxe **facile à lire et à écrire**.*

Un document balisé par Markdown peut être lu en l'état sans donner l'impression d'avoir été balisé ou formaté par des instructions particulières.

*Un document balisé par Markdown peut être converti en **HTML**, en **PDF** ou en d'autres formats."* Wikipédia

Le nom du fichier est suivi de l'extension **.md**

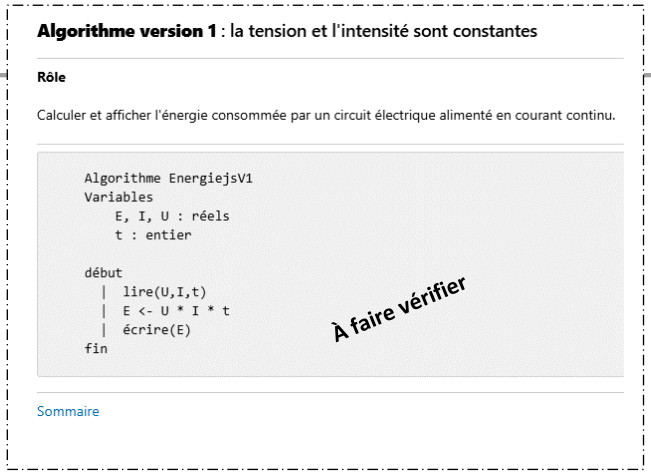
Markdown 1. Documentez le projet dans le fichier ***doc1.md*** avec :
- Un titre (voir ci-contre).
- Un descriptif (uniquement le rôle dans la spécification).
- L'algorithme fourni.

- Pour rédiger en **Markdown, consultez :** https://webge.fr/md

Résultat attendu

Pour voir le résultat dans VSCode : cliquez sur en **haut et à droite** de la fenêtre ou *F1 -> Markdown:Ouvrir l'aperçu sur le côté.*

- **Complétez** le fichier *index.md* et testez-le dans VSCode.



Algorithme version 1 : la tension et l'intensité sont constantes

Rôle

Calculer et afficher l'énergie consommée par un circuit électrique alimenté en courant continu.

```
Algorithme EnergiejsV1
Variables
    E, I, U : réels
    t : entier

début
  |  lire(U,I,t)
  |  E <- U * I * t
  |  écrire(E)
fin
```

Sommaire

À faire vérifier

2.3 Programme version 2 : changement des unités de E

Objectifs : analyser un problème et proposer un algorithme, coder une structure alternative.

Spécification (seuls changements de EnergiejsV2 par rapport à la V1 ; les entrées et les préconditions restent identiques) :
Nom : EnergiejsV2
Rôle : on affiche E dans l'unité adaptée à sa valeur.
Donnée de sortie : on ajoute l'unité de E, « J » ou « Wh ».
Traitement : après E ← U × I × t, on ajoute une structure alternative.

- si 0 ≤ E ≤ 360 (0,1 Wh) alors écrire E en J
- sinon écrire E en Wh

Postcondition : E est affichée en J si E ≤ 0,1 Wh, sinon en Wh.

A. Algorithmique

Algorithme 1. Écrivez l'algorithme EnergiejsV2 sur le **DR1**.

Remarque : EnergiejsV2 doit répondre à la spécification ci-dessus tout en conservant la structure de l'algorithme *EnergiejsV1*.

B. Codage en JavaScript

Implémentez l'algorithme **EnergiejsV2** dans le fichier **index.html** du répertoire **v2**.

Ressources** : **annexe 1, §3 et §5

Pour en savoir plus :

- *if…else :* MDN, if…else
- *L'opérateur logique && :* MDN, ET logique

C. Amélioration à apporter au code

Affichez le résultat en mWh si 360 J < E ≤ 3600 J (0,1 Wh < E ≤ 1 Wh)

Ressources** : **annexe 1, §4

Pour en savoir plus : Méthode toFixed() : MDN, toFixed()

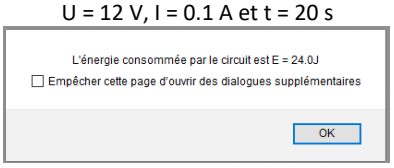
D. Documentation en Markdown

Markdown 2.

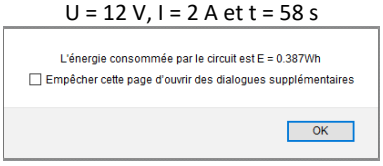
- **Documentez** le projet dans le fichier ***doc2.md*** avec :
  - Un titre : Algorithme version 2 : changement des unités de E.
  - Un descriptif (uniquement le rôle dans la spécification).
  - L'algorithme correspondant au programme que vous avez réalisé dans le **paragraphe C**.
- **Complétez** le fichier *index.md* et testez-le dans VSCode.

Exemples de résultat attendu (jeux d'essai)

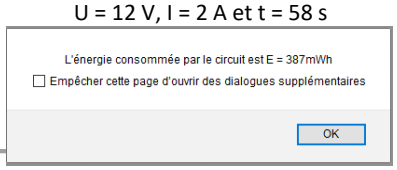
U = 12 V, I = 0.1 A et t = 20 s



U = 12 V, I = 2 A et t = 58 s

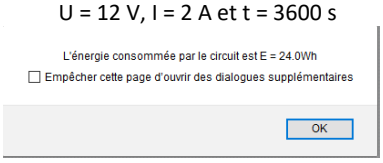


U = 12 V, I = 2 A et t = 58 s



À faire vérifier

U = 12 V, I = 2 A et t = 3600 s



2.4 Programme version 3 : le programme précédent devient une fonction

Objectif : être capable de créer une fonction personnalisée et de rédiger un appel de fonction.

Spécification (seuls changements de EnergiejsV3 par rapport à la V2)
Nom : EnergiejsV3
Structure : le traitement de la V2 (lire, calculer, choisir l'unité, écrire) devient une fonction calculE.

A. Codage en JavaScript

- **Copiez** le code de la page **index.html** située dans **v2** dans la page index.html située dans **v3.**
  ***Ressources** : **annexe 1, §6** pour l'écriture des fonctions*
  Pour en savoir plus : MDN, fonctions.
- **Intégrez** le code du programme précédent dans une fonction que vous nommerez ***calculE.***
- **Écrivez le code d'appel de *calculE***.
- **Placez** le code JavaScript dans un fichier **script.js** à créer dans le répertoire **scripts** de **v3**.
- **Modifiez la balise script dans index.html.**

Résultat attendu

Le fonctionnement doit rester le même que celui obtenu avec le programme **V2**.

À faire vérifier

2.5 Programme version 4 : formulaire et évènements

Objectif : être capable de concevoir un formulaire simple et de gérer des clics de souris (évènements).

Spécification (seuls changements de EnergiejsV4 par rapport à la V3)

Nom : EnergiejsV4
Rôle : le calcul se fait maintenant dans une page web munie d'un formulaire. On ne passe plus par des boîtes de dialogue.
Données d'entrée : U, I et t sont lus dans trois champs texte du formulaire (lire(U, I, t) devient la lecture des champs), toujours convertis en nombres.
Donnée de sortie : E est écrite dans un quatrième champ du formulaire, en lecture seule.
Déclenchement : le traitement est placé dans la fonction calculer(), sans paramètre ni valeur renvoyée. Elle s'exécute à chaque clic sur le bouton « Calculer » (évènement onclick), et non plus une seule fois à l'ouverture de la page. On peut donc refaire plusieurs calculs sans recharger la page.
Traitement (rappel) : on calcule E ← U × I × t en J, puis :
- si 0 ≤ E ≤ 360 : E en J
- sinon si 360 < E ≤ 3600 : E en mWh
- sinon : E en Wh

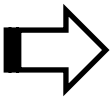
Précondition ajoutée : les trois champs sont remplis avec des nombres avant le clic. Sinon, le champ E affiche NaN.

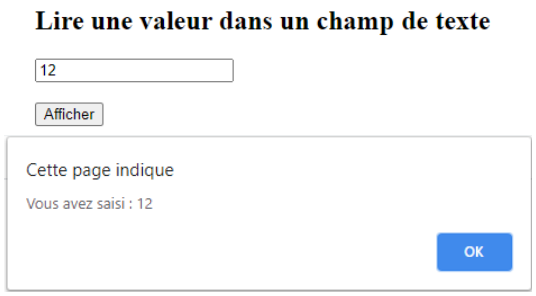
A. Codage

Résultats attendus (jeux d'essai)

- Écrivez le code de l'**annexe 2** dans le fichier ***exemple.html*** situé dans **v4** et testez-le dans un navigateur.

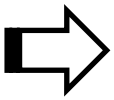
Ressource : annexe 1, §9
Pour en savoir plus :
- formulaire : MDN, premier formulaire
- getElementById() : MDN, getElementById()

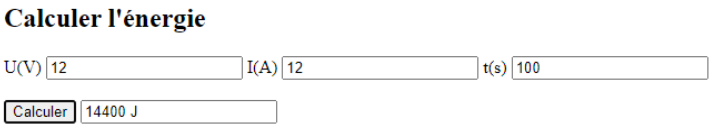




- En vous inspirant de l'**annexe 2**, codez le formulaire et la fonction de calcul de l'énergie : **calculer().**

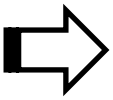
Ressource : annexe 1, §9





- Modifiez le code de la fonction **calculer()** pour que la valeur de l'énergie s'affiche comme ci-contre.

Ressource : annexe 1, §9





À faire vérifier

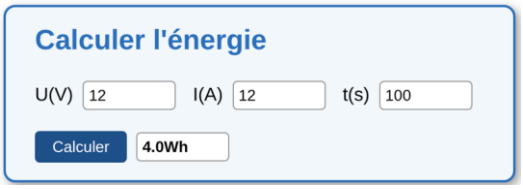
- **Placez** le code JavaScript dans un fichier **script.js** dans le répertoire **scripts** de **v4**.

B. Design de la boîte de calcul

- Le fichier style.css situé dans le dossier styles contient le code CSS de la figure ci-contre.

⚠ **Ce fichier ne doit pas être modifié.**

Ajoutez le lien vers ce fichier et testez le bon fonctionnement de la boîte.
Si nécessaire, modifiez le fichier html pour obtenir le rendu ci-contre.



À faire vérifier

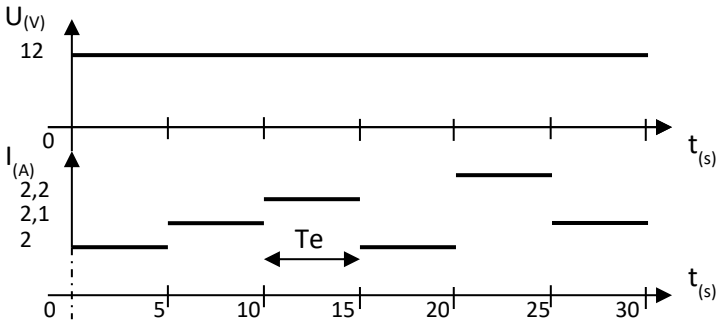
2.6 Programme version 5 : la tension est constante, l'intensité varie dans le temps

Objectif : être capable d'écrire une structure répétitive pour lister le contenu d'un tableau, instrumenter son code.

A. Présentation

Les grandeurs électriques ont été supposées constantes dans les versions précédentes. Dans de nombreux cas, si la tension peut être considérée comme constante, l'intensité du courant varie dans le temps.
Pendant chaque période Te, on considère que l'intensité garde la valeur mesurée I[k] : l'énergie consommée pendant cette période vaut **U × I[k] × Te**. L'énergie totale est la **somme** de ces énergies : **E = U × I[0] × Te + U × I[1] × Te + …**

Exemple : la tension U est constante, I varie, la période de mesure (échantillonnage) nommée **Te = 5s**



k	I[k]
0	2
1	2,1
2	2,2
3	2
4	2,3
5	2,1

B. Analyse du programme

On donne le programme ci-dessous.

```
1.  let U = 12;
2.  let I = [2,2.1,2.2];
3.  let Te = 5; // échantillonnage toutes les 5 s
4.  let E = 0;
5.  for (let k = 0; k < I.length; k++) {
6.      E = E + I[k]*U*Te;
7. }
```

Ressource : annexe 1, §7 et §8
Pour en savoir plus :
- Tableau : MDN, tableaux
- Boucle for : MDN, boucle for

Travail demandé

À l'aide du tableau du document réponse 2, déterminez la valeur de E en J et en mWh à la sortie de la boucle de ce programme.

Répondez sur le DR2.

C. Codage

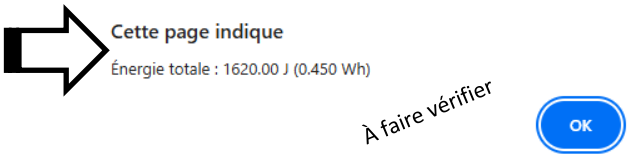
Travail demandé

- **Créez** une nouvelle fonction de calcul de E que vous nommerez ***calculE2*** dans v5/scripts/script.js avec :
  - les valeurs de $I_{(A)}$ contenues dans un tableau.
  - $U_{(V)}$ et $Te_{(s)}$ à entrer avec des boîtes de dialogue.
- **Donnez** le résultat en J et en Wh dans une boîte de dialogue.

Exemple de résultat attendu (jeux d'essai)

Pour **U = 12, Te = 5** et

I = [2,2.1,2.2,2,2.3,2.1,2.5,2.5,2.5,1.8,1.7,1.7,1.6]



À faire vérifier

INSTRUMENTATION DU CODE

Principe : on ajoute au programme des instructions qui ne servent pas au calcul. Elles affichent la valeur des variables pendant l'exécution, pour vérifier que le programme fait bien ce qu'on attend et trouver une erreur. C'est l'équivalent d'un voltmètre placé en un point d'un circuit.

En JavaScript, on utilise console.log(). L'affichage se fait dans la console du navigateur (F12, onglet Console) et pas dans la page, donc l'utilisateur ne le voit pas. Contrairement à alert(), l'exécution ne s'arrête pas.

Pour suivre le détail du calcul dans la console du navigateur **INSTRUMENTEZ** le code en écrivant **console.log("k = " + k + " E = " + E);** dans la boucle for, après E = E + I[k]*U*Te;

DEBUG [F12] dans le navigateur

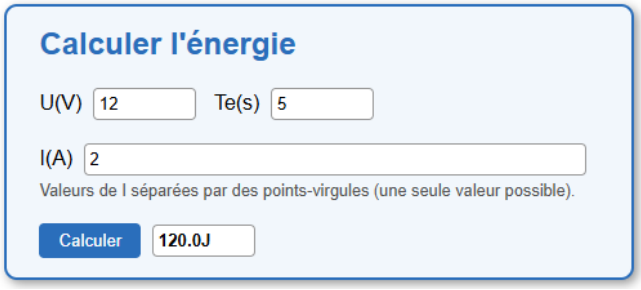
3 Synthèse

Présentation

Le dossier « synthese » contient les fichiers nécessaires à la création d'une interface destinée à calculer l'énergie E, que le courant I soit constant ou variable dans le temps.

1. Complétez le fichier script.js.
2. Vérifiez le bon fonctionnement avec les jeux de test (U = 12 V, Te = 5 s) :

- « 2 » ou « 2; » : 120.0 J
- « 2,5 » : 150.0 J
- « 1; 2; » : 180.0 J
- champ vide : Saisie incorrecte
- série du sujet : 450 mWh



À faire vérifier

Pour aller plus loin…

- **Gestion des erreurs**

```
// Ajouter une validation des entrées
    let U = parseFloat(prompt("Tension U (V) :"));
    if (isNaN(U) || U <= 0) {
        alert("Erreur : la tension doit être un nombre positif !");
    }
```

- **Améliorer l'UX du formulaire :**

```
// Effacer le résultat quand on modifie les valeurs
// addEventListener("input", …) exécute la fonction à chaque modification du champ U
    document.getElementById("tension").addEventListener("input", function() {
    document.getElementById("energie").value = "";
});
```

Annexe 1 - Résumé JavaScript

*Les notions utilisées dans ce TP (complément dans **ressource.md**). Documentation complète : **MDN, guide JavaScript** (lien dans les ressources, page 2).*

1. Variables

```
let U = 12;          // nombre
let nom = "moteur";  // texte (chaîne)
let I = [2, 2.1];    // tableau
```

let déclare une variable (**var** : ancienne écriture, à éviter).
Le nom respecte la casse : E et e sont deux variables différentes.
// introduit un commentaire.

2. Entrées et sorties

```
U = parseFloat(prompt("Tension U en V"));
alert("E = " + E + " J");
console.log("E = " + E);   // console (F12)
```

prompt() renvoie toujours du **texte**.

parseFloat() le convertit en nombre flottant

parseInt() le convertit en nombre entier.

+ entre un texte et une valeur **concatène** (met bout à bout).

3. Opérateurs

```
+  -  *  /          calcul
<  <=  >  >=  ===  !==   comparaison
&&  (ET)   ||  (OU)   !  (NON)
```

= affecte une valeur ; === compare deux valeurs.

4. Arrondi

```
E.toFixed(1)   // 386.666… → "386.7"
```

Renvoie le nombre arrondi avec le nombre de décimales demandé.

5. Structure alternative

```
Exemple

if (nb>0 && nb<=10) {
    alert(nb + " appartient à ]0,10]");
} else if (nb > 10 && nb <= 100) {
    alert(nb + " appartient à ]10,100]");
} else {
    alert(nb + " > 100");
}
```

Un seul bloc { } est exécuté : le premier dont la condition est vraie.

6. Fonction

```
function calculE() {
    let E = U * I * t;
    alert(E);
}
calculE();          // appel
```

Une fonction ne s'exécute que lorsqu'on l'**appelle**.

7. Tableau

```
let I = [2, 2.1, 2.2];
I[0]       // 2 : 1er élément (indice 0)
I[2]       // 2.2
I.length   // 3 : nombre d'éléments
```

Les indices commencent à **0** : le dernier élément est `I[I.length - 1]`.

8. Boucle for

```
for (let k = 0; k < I.length; k++) {
    E = E + I[k] * U * Te;
}
```

Départ `k = 0` ; on répète tant que `k < I.length` ; k++ ajoute 1 à k après chaque tour.

9. JavaScript dans la page web

```
<script src="scripts/script.js"></script>
<input type="text" name="tension" id="tension">
<input type="button" value="Calculer"
       onclick="calculer()">
// U ← Valeur saisie dans le formulaire
let U = parseFloat(
    document.getElementById("tension").value);
// Champ du formulaire ← E
document.getElementById("energie").value = E;
```

Remarque

`getElementById("tension")` désigne l'élément d'id tension ;

`.value` est son contenu (du texte).

`onclick` appelle la fonction lors d'un clic sur le bouton.

10. Erreurs fréquentes

- `"12" + "2"` donne `"122"` : penser à `parseFloat()`.
- `if (E = 0)` affecte au lieu de comparer : écrire `===`.
- Une accolade { ouverte doit être fermée par }.
- `getElementByID` n'existe pas : respecter la casse (`getElementById`).
- Pour trouver une erreur : touche **F12**, onglet **Console**.

Annexe 2 – Exemple de traitement d'un formulaire en JavaScript

```
<!DOCTYPE html>
<html>

<head>
    <meta charset="utf-8">
    <title>Lire une valeur dans un champ de texte</title>

    <script>
        function afficher() {
            let saisie = document.getElementById("entree").value;
            alert("Vous avez saisi : " + saisie);
        }
    </script>
</head>

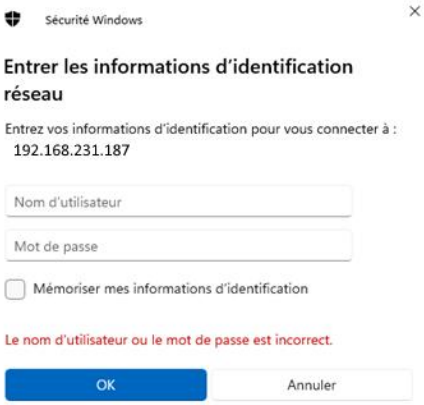
<body>
  <div id="boite">
    <h2>Lire une valeur dans un champ de texte</h2>
    <form>
        Valeur <input type="text" id="entree" ><br><br>
        <input type="button" id="bouton" value="Afficher" onclick="afficher()">
    </form>
  </div>
</body>
</html>
```

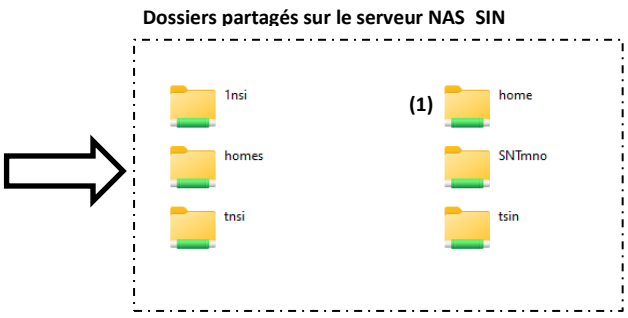
Annexe 3 - Installation du dossier du TP dans votre compte sur le serveur Nas_SIN

1. Connexion au serveur NAS_SIN pour accéder au dossier partagé du cours.



- Sélectionnez  et entrez **\\192.168.231.187** dans la fenêtre Exécuter.
- Complétez la fenêtre de sécurité Windows représentée ci-dessous avec l'**identifiant** et le **mot de passe** de votre compte sur le serveur **NAS_SIN pour accéder aux dossiers partagés.**





(1) Votre dossier personnel

- Double-cliquez sur le dossier **du cours.**

2. Dossier à utiliser dans le TP

- Copiez le dossier **WEB6a** situé dans ***DOSSIER_du_COURS/5_IHM_WEB/*** et collez-le dans ***home/www/***

