Installation and use of Ornement

Ornement is a software solution for **enriching text using tags**. The advantage of Ornement is that it is **easy to install and use**. The purpose of this document is to describe how to use Ornement using all the available tags. The javascript standard used is **ES6**.

Installation

There are two ways to install Ornement:

- You can install Ornement using the **npm install ornement** command.
- Here are the different steps for installing Ornement in your project:
    - Download Ornement from its repository into the folder you want, which may be in your project.
    - That's it, you only need to complete one step to install Ornement!

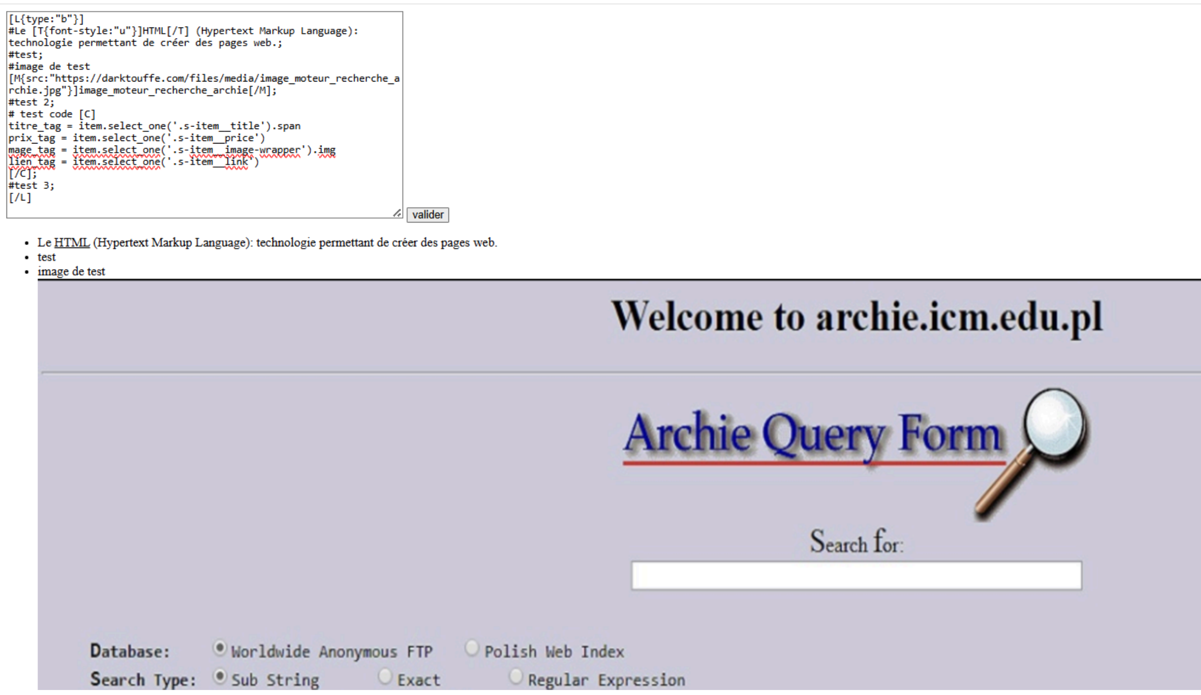
Use

Demo

You can quickly see how Ornement works using the project available on GitHub, using Node-CLI, or using the React project available in Demo > ornement_react_demo:

- If you installed Ornement directly from git via git clone:
    - Using a terminal, install http-server (npm install http-server), then launch the server (http-server) (You can also use Live Server from VS Code if you are more comfortable with it).
    - Open your browser and type http://localhost:8080.

- Go to the folder where you installed Ornement.
- Then go to Demo and open testForm.html.
- In the text box, enter the different test sets below. Here is an example:



```
[L{type:"b"}]
#Le [T{font-style:"u"}]HTML[/T] (Hypertext Markup Language):
technologie permettant de créer des pages web.;
#test;
#image de test
[M{src:"https://darktouffe.com/files/media/image_moteur_recherche_a
rchie.jpg"}]image_moteur_recherche_archie[/M];
#test 2;
# test code [C]
titre_tag = item.select_one('.s-item__title').span
prix_tag = item.select_one('.s-item__price')
mage_tag = item.select_one('.s-item__image-wrapper').img
lien_tag = item.select_one('.s-item__link')
[/C];
#test 3;
[/L]
```

valider

- Le <u>HTML</u> (Hypertext Markup Language): technologie permettant de créer des pages web.
- test
- image de test

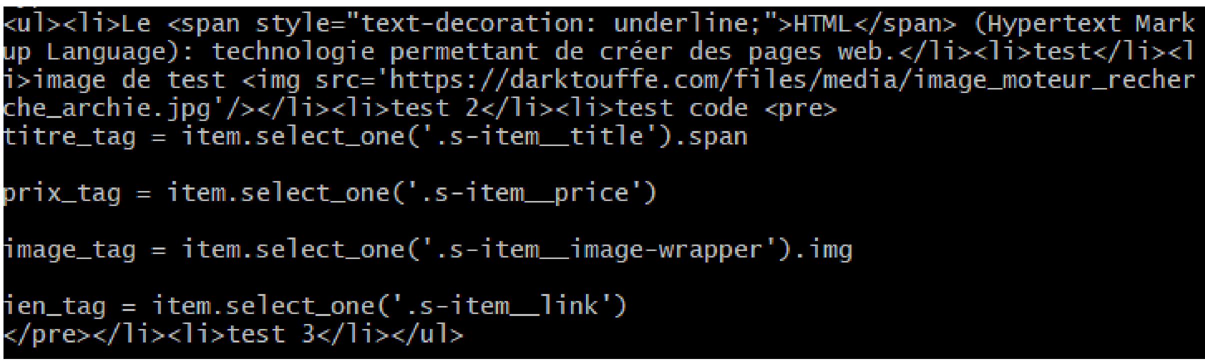
Welcome to archie.icm.edu.pl
Archie Query Form
Search for:
Database: Worldwide Anonymous FTP Polish Web Index
Search Type: Sub String Exact Regular Expression

- This is how to use Ornement with vanilla JS.

- If you are using Node-CLI directly
  - Create a test folder and then a JavaScript file (here test.js).
  - Enter the following code. You can replace the content of input with one of the different test sets below:

```
// test.js
import { Ornement } from 'ornement';

const input = `[L{type:"b"}]

#Le [T{font-style:"u"}]HTML[/T] (Hypertext Markup Language): technologie permettant de créer des pages web.;

#test;

#image de test [M{src:"https://darktouffe.com/files/media/image_moteur_recherche_archie.jpg"}]image_moteur_recherche_archie[/M];

#test 2;

# test code [C]

titre_tag = item.select_one('.s-item__title').span

prix_tag = item.select_one('.s-item__price')

image_tag = item.select_one('.s-item__image-wrapper').img

ien_tag = item.select_one('.s-item__link')

[/C];

#test 3;

[/L]`;

const result = Ornement(input);
console.log(result);
```

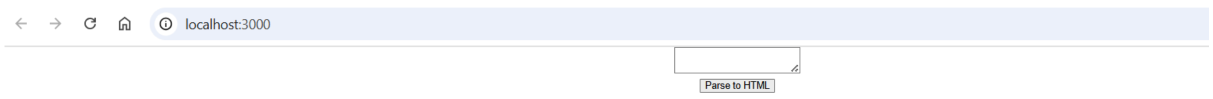
  - Run the command node test.js. You should get a result like this:



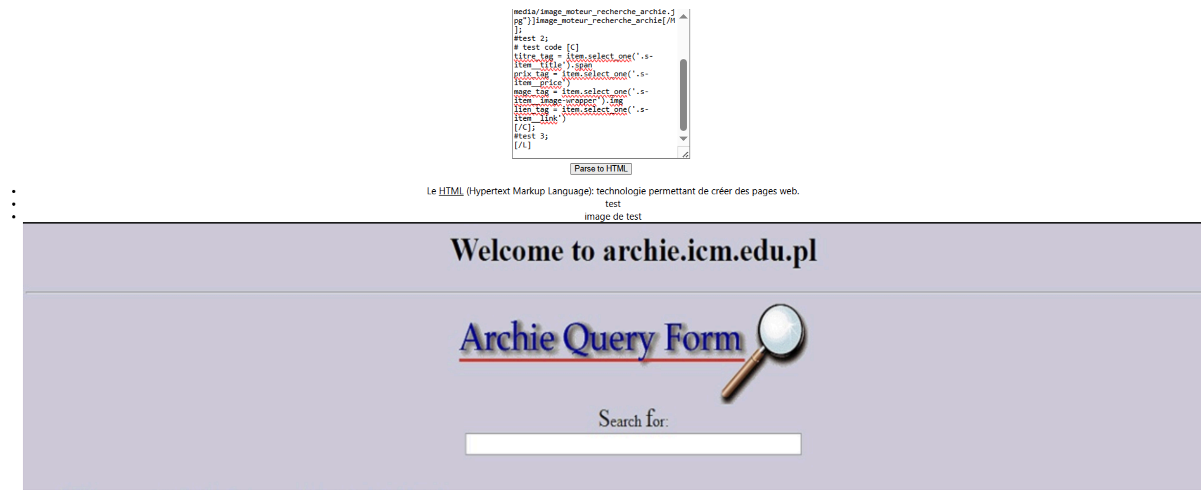
```
<ul><li>Le <span style="text-decoration: underline;">HTML</span> (Hypertext Mark
up Language): technologie permettant de créer des pages web.</li><li>test</li><l
i>image de test <img src='https://darktouffe.com/files/media/image_moteur_recher
che_archie.jpg'/></li><li>test 2</li><li>test code <pre>
titre_tag = item.select_one('.s-item__title').span

prix_tag = item.select_one('.s-item__price')

image_tag = item.select_one('.s-item__image-wrapper').img

ien_tag = item.select_one('.s-item__link')
</pre></li><li>test 3</li></ul>
```

- This is how you would use Ornament in a project using Node (such as React, for example).

- If you want to use the React project, you need to:
  - Go to Demo > ornement_react_demo using a terminal, then run the npm start command.
  - Next, go to http://localhost:3000 using your browser. You should see a page like the one below. Enter the test games in the text box.



  - Here is the result:



```
media/image_moteur_recherche_archie.j
pg"}]image_moteur_recherche_archie[/M
];
#test 2;
# test code [C]
titre_tag = item.select_one('.s-
item__title').span
prix_tag = item.select_one('.s-
item__price')
mage_tag = item.select_one('.s-
item__image-wrapper').img
lien_tag = item.select_one('.s-
item__link')
[/C];
#test 3;
[/L]
```

Le HTML (Hypertext Markup Language): technologie permettant de créer des pages web.
test
image de test

Here are the different test sets:

- [T{style:"t2", font-style:"u"}]Introduction[/T]
- [M{src:"https://es.wikipedia.org/static/images/icons/wikipedia.png"}]Wikipedia Logo[/M]
- [T{link:"https://fr.wikipedia.org/wiki/Ada_Lovelace"}]https://fr.wikipedia.org/wiki/Ada_Lovelace[/T]
- Différents éléments dans une liste:

- [L{type:"b"}]

  #Le [T{font-style:"u"}]HTML[/T] (Hypertext Markup Language): technologie permettant de créer des pages web.;

  #test;

  #test code [C]
  titre_tag = item.select_one('.s-item__title').span

  prix_tag = item.select_one('.s-item__price')

  mage_tag = item.select_one('.s-item__image-wrapper').img

  ien_tag = item.select_one('.s-item__link')
  [/C];

  #test 3 <test>;

  [/L]

- [L{type:"b"}]

  #Le [T{font-style:"u"}]HTML[/T] (Hypertext Markup Language): technologie

  permettant de créer des pages web.;

  #test;

  #image de test

  [M{src:"https://darktouffe.com/files/media/image_moteur_recherche_archie.jpg"}]i

  mage_moteur_recherche_archie[/M];

  #test 2;

  # test code [C]

  titre_tag = item.select_one('.s-item__title').span

  prix_tag = item.select_one('.s-item__price')

  mage_tag = item.select_one('.s-item__image-wrapper').img

  lien_tag = item.select_one('.s-item__link')

  [/C];

#test 3;

[/L]

The Engine

The engine consists of the **Ornement.js** file and the **Parsers** folder. This folder is itself made up of various **parsers** whose purpose is to **generate html** code from the Ornement tags that are used.

Integrating Ornement into your project

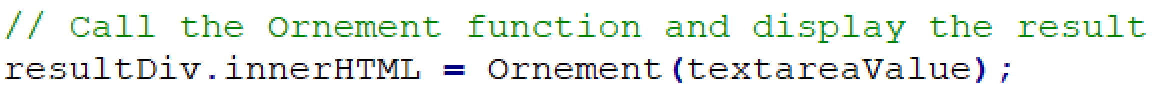
To integrate Ornament into your project, as seen in the demo, there are two ways to do so

Vanilla JS

To integrate Ornement into your project, you need to install the folder in your project as mentioned in Installation (above).
A tag is first used in an input or textarea field **(1)**. You then need to import Ornement and retrieve the content of this field **(2)**. Finally, you pass the recovered content to the Ornement engine, which generates and returns the html tags **(3)**.

We're going to take a detailed look at how to use Ornement in Javascript (**ES6 and later versions**). I'll be using the files in the Demo folder, which are available to you:

1. Create an input or textarea field, and assign it an ID, in this case Ornement.
   ```
   <textarea id="Ornement"></textarea>
   ```
   Create a tag which will be used to store the tags obtained using Ornement, in this case I'm creating a div tag, with result as the ID.
   ```
       <div id="result">
       </div>
   ```
2. Import Ornement and retrieve the data:
   a. Import the Ornement engine as follows (adapt the file path to your situation).
   ```
   import { Ornement } from "../Ornement.js";
   ```
   b. Retrieve the data as follows.
   ```
   const textareaValue = document.getElementById("Ornement").value;
   ```
3. Generate the tags as follows:
   a. create a variable whose purpose is to be linked to the div tag with the result ID.
   ```
   const resultDiv = document.getElementById("result");
   ```
   b. Pass the recovered content (textareaValue) into Ornement, then integrate the result returned by Ornement into the result div.



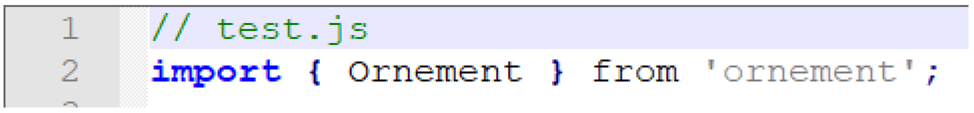
```
// Call the Ornement function and display the result
resultDiv.innerHTML = Ornement(textareaValue);
```

The result should now be displayed in your tag!

Feel free to use the files in the Demo folder.

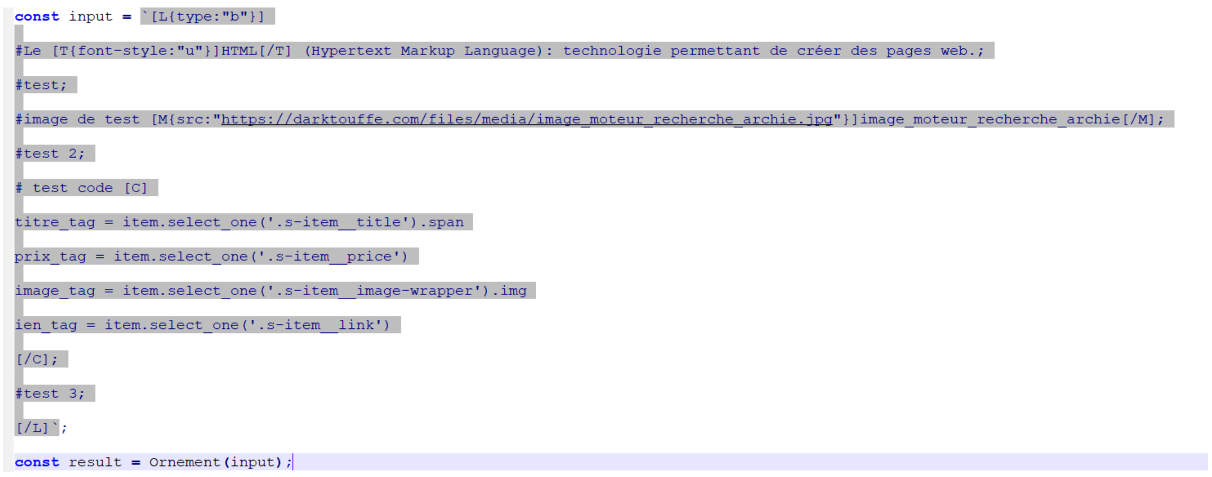
Node Project

After installing Ornement with npm install ornement, you simply need to import Ornement as shown in the demo:



```
// test.js
import { Ornement } from 'ornement';
```

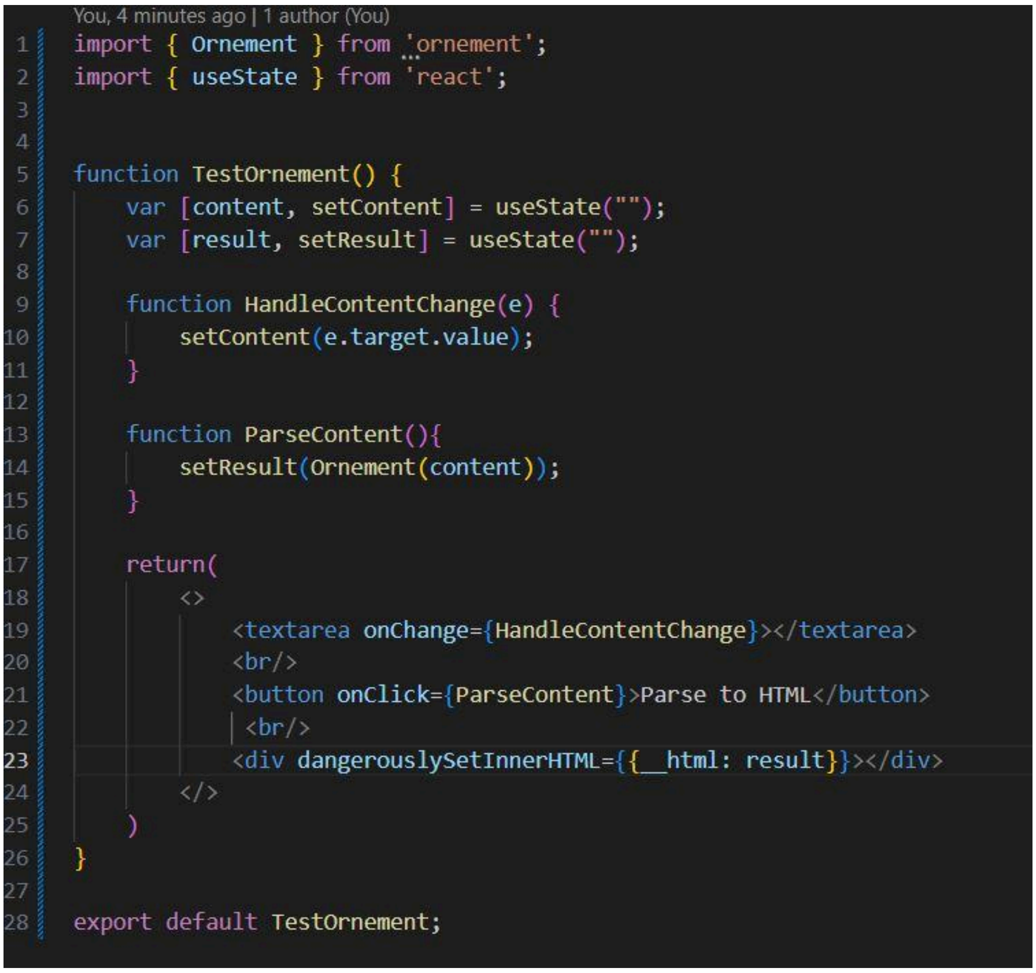
You can then use Ornament in the same way as in the demo, saving the result in a variable, for example.



```
const input = `[L{type:"b"}]
#Le [T{font-style:"u"}]HTML[/T] (Hypertext Markup Language): technologie permettant de créer des pages web.;
#test;
#image de test [M{src:"https://darktouffe.com/files/media/image_moteur_recherche_archie.jpg"}]image_moteur_recherche_archie[/M];
#test 2;
# test code [C]
titre_tag = item.select_one('.s-item__title').span
prix_tag = item.select_one('.s-item__price')
image_tag = item.select_one('.s-item__image-wrapper').img
ien_tag = item.select_one('.s-item__link')
[/C];
#test 3;
[/L]`;
const result = Ornement(input);
```

React

To use Ornement with React, start by creating a new project with React using the command **npx create-react-app ornement_react_demo**. Install Ornement with npm install ornement. Then, you can create a TestOrnement component as shown below:



```
You, 4 minutes ago | 1 author (You)
import { Ornement } from 'ornement';
import { useState } from 'react';


function TestOrnement() {
    var [content, setContent] = useState("");
    var [result, setResult] = useState("");

    function HandleContentChange(e) {
        setContent(e.target.value);
    }

    function ParseContent(){
        setResult(Ornement(content));
    }

    return(
        <>
            <textarea onChange={HandleContentChange}></textarea>
            <br/>
            <button onClick={ParseContent}>Parse to HTML</button>
            <br/>
            <div dangerouslySetInnerHTML={{__html: result}}></div>
        </>
    )
}

export default TestOrnement;
```

The code below performs the following steps:

1. We start by importing Ornement.
2. Inside the component, we create a text area. This text area will be linked to the **handleContentChange event**, which saves any changes to the text area in the content state.
3. Next, we create a button linked to an **Onclick event**, with **ParseContent** as the callback, which will **translate the content and change the result state**. We create a div where the result will be displayed.

You can then integrate this component wherever you want.

Using the different tags

Here is a general example of the use of a tag, identified here by a base S (so it doesn't exist!):

- **[S{attribute1:”value1”, etc.}]Text[/S]** to add rich text, a list, code or an image.
  - **S** corresponds to the tag.
  - **[S...]** is the opening tag used.
  - **{attribute1: ‘value1’, etc.}** are the attributes of the tag.
  - **Text** is the text of the tag.
  - **[/S]** is the closing tag used.
- An escape character will be provided so that the symbols can be used freely.

/!\ between **attribute1**: and **‘value1’** no space is allowed!

The rest of this document describes the various tags that can be used, each of which can be tested using the testForm.html file.

The T tag

This tag can be used to enrich text of any length. It is used in the following way:
[T{attribute1:”value1”, etc.}]Text[/T]

The different attributes allow the tag to be used in different ways:

- **font-style:** Used to style the text. Possible values are:
  - **i**: italic, [T{font-style:"i"}]Text[/T].
  - **b**: bold, [T{font-style:"b"}]Text[/T].
  - **u**: underlined, [T{font-style:"u"}]Text[/T].
  - **s**: barré, [T{font-style:"s"}]Text[/T].
  - You can use all or some of these elements together, for example : [T{font-style:"bui"}]Text[/T].
- **link**: Inserts a link in the text, [T{link:"https://fr.wikipedia.org/wiki/Ada_Lovelace"}]https://fr.wikipedia.org/wiki/Ada_Lovelace[/T].
- **color**: Used to specify a colour in hexadecimal format only, [T{color:"#00FF00"}]Texte[/T] (will give a green text).
- **highlight-color**: Allows you to set a highlight colour in hexadecimal format only, [T{highlight-color:"#00FF00"}]Texte[/T] (will give text underlined in green).
- **style**: Used to style the text. Here are the possible values:
  - **tx**: For your text to be understood as a title, x can be from 1 to 6, [T{style:"t3"}]Sources:[/T], [T{style:"t2"}]Sources:[/T], [T{style:"t5"}]Sources:[/T].
  - sup: So that your text is in clue form, test[T{style:"sup"}]2[/T].
  - sub: To make your text superscript, test[T{style:"sub"}]2[/T].

You can combine the various attributes. For example, if you want text in red, highlighted in yellow, bold and underlined, you would use [T{color:"#FF0022", highlight-color:"#FFFF00",font-style:"ub"}]Lots of cumulative attributes[/T]. You can put spaces, after the comma, between attributes.

The L tag

This tag is used to create bulleted or numbered lists. It is used in the following way:
[L{type:”type”}]
#element1;
#element2;
…

#element3;
[/L]

Here are the criteria and rules to follow:

- ‘type’ must always be specified, it can take the empty type “”, bullet point “b”, or numbered “n”. The empty type corresponds by default to the bullet point type:
  - Bullet point type:
    [L{type:"b"}]
    #élément 1;
    #élément 2;
    #élément 3;
    [/L]

    ou

    [L{type:""}]
    #élément 1;
    #élément 2;
    #élément 3;
    [/L]

  - Numbererd type:
    [L{type:"n"}]
    #élément 1;
    #élément 2;
    #élément 3;
    [/L]

- A line feed is required between the opening tag and the first element of the list, between each element of the list, and between the last element of the list and the closing tag.
- each list item starts with a diez “#” and ends with a semicolon “;”, between the diez and the item or between the item and the semicolon, there can be many spaces.

The M tag

The M tag can be used to include online media, such as pictures, videos or audio. It is used as follows:
[M{src:”lien_media”}]Text[/M]

Here are the characteristics of this tag:

- The src attribute is used to enter the media address, which must be on the Internet.
- The text between the M tags is not compulsory, but it will help you remember the image you have set in edit mode.
- The following types of media are authorized:
  - 'jpg', 'jpeg', 'png', 'gif', 'bmp', ‘webp’ for pictures.
  - 'mp3', 'wav', 'ogg', 'aac' for audios.

    - 'mp4', 'webm', 'avi', 'mkv'for videos.

The C tag

The C tag is used to include stylized code. It has no attributes. It is used as follows:

[C]
Code
[/C]

Skip a line after the opening tag, and skip a line at the end of the written code.

Ornement and XSS vulnerabilities.

Ornement provides basic protection against XSS vulnerabilities by transforming the chevrons < into &lt; and > into &gt. However, basic protection against XSS vulnerabilities must be put in place. You are therefore advised to use software such as DOMPurify or an equivalent.

For all other security flaws such as SQL injections, the necessary defences must also be put in place. **Ornement is not responsible for any security problems related to its use.**