Documentazione: *utils.py*

In questa breve (più o meno) documentazione troverete definizioni ed esempi di una libreria che abbiamo creato per voi.
pyscript permette di inserire anche codice JavaScript Plane, utile per l'interazione con il DOM, ma considerate le finalità dell'esercizio, abbiamo creato questa libreria come *layer* (strato) aggiuntivo tra le funzioni JS che pyscript vi permette di utilizzare e il codice che creerete contenente la logica del vostro esercizio. Non dovrete utilizzare codice JS su *your-solution.py*, ma utilizzerete solo python, e richiamerete i relativi metodi della libreria per ottenere gli effetti JS che vi illustreremo di seguito.

Import

Innanzitutto, importiamo la classe Utils nel nostro codice, così da poter richiamarne i metodi:

```
import pyodide
import js
from utils import Utils

my_utils = Utils(pyodide, js)
```

Una volta importata la classe questa sarà disponibile dentro l'oggetto nel quale la abbiamo importata. Ad esempio, scriveremo qualcosa del tipo *my_utils.method,* dove *method* sarà uno di quelli sotto elencati

Metodi disponibili

SE SIETE CURIOSI…
Nel file utils.py sono presenti le funzioni sotto descritte, ma ai fini della programmazione ad oggetti, i metodi sono dichiarati con un campo supplementare (*self*) che si rifà al costruttore.

La parola chiave *self* è necessaria ai fini della costruzione dei metodi, ma voi utilizzatori, nel richiamare un metodo, potete ignorarla (python vuole che i metodi siano definiti in questo modo e richiamerà *self* in maniera autonoma ogni volta che un metodo viene utilizzato).

Pertanto, **qui sotto trovate i metodi esattamente come devono essere richiamati** in *your-solution.py*.

➢ writeToConsole

```
writeToConsole(value)
```

Funzionamento:

Scrive il contenuto di **value** in console sul browser.

Esempio:

Python:

```
writeToConsole("Boolean")
```

Console:

```
>> Boolean
```

➢ getHtmlElement

```
getHtmlElement(id)
```

Funzionamento:

La funzione restituisce un elemento selezionato nel DOM con id uguale al valore di **id** passato.

Esempio:

HTML:

```
<div id= "name"> Boolean </div>
```

Python:

```
selected = getHtmlElement("name")
writeToConsole(selected.innerHtml)
```

Console:

```
>> Boolean
```

➢ writeToHtmlElement

```
writeToHtmlElement(element, text)
```

Funzionamento:

La funzione scrive il contenuto di **text** nell'innerHtml dell'oggetto DOM contenuto in **element**

Esempio:

HTML:

```
<div id= "name"> </div>
```

Python:

```
element = getHtmlElement("name")
writeToHtmlElement(element, "Boolean")
```

HTML (result):

```
<div id= "name" > Boolean </div>
```

➢ emptyInputElement

```
emptyInputElement(element)
```

Funzionamento:

La funzione pulisce il campo value di un oggetto DOM <input> contenuto in **element**

Esempio:

HTML:

```
<input id= “input_field” value= “3”> </input>
```

Python:

```
element = getHtmlElement(“input_field”)
emptyInputElement(element)
```

HTML (result):

```
<input id= “input_field” value= “”> </input>
```

➢ disableInputElement

```
disableInputElement(element)
```

Funzionamento:

La funzione setta il flag *disabled = true* all’oggetto DOM contenuto in **element**

Esempio:

HTML:

```
<input id= “name”> </input>
```

Python:

```
element = getHtmlElement(“name”)
disableInputElement(element)
```

HTML (result):

```
<input id= “name” disabled> </ input >
```

➢ addOnClickEventToHtmlElement

```
addOnClickEventToHtmlElement (element, event)
```

Funzionamento:

La funzione aggiunge un evento al *onClick* dell'oggetto DOM contenuto in **element**. In **event** deve essere passata la funzione da associare al click

Esempio:

HTML:

```
<div id= "name"> Boolean </div>
```

Python:

```
def print_func():
      writeToConsole("done!")

element = getHtmlElement("name")
addOnClickEventToHtmlElement (element, print_func)
```

Console (al click di):

```
>> done!
```

➢ removeOnClickEventFromHtmlElement

```
removeOnClickEventFromHtmlElement (element)
```

Funzionamento:

La funzione disabilita eventuali azioni al click dell'elemento DOM contenuto in **element,** settando onClick = false

Esempio:

HTML:

```
<div id= “name” onClick= “print_func()”> Boolean </div>
```

Python:

```
element = getHtmlElement(“name”)
removeOnClickEventFromHtmlElement (element)
```

HTML (result):

```
<div id= “name”> Boolean </div>
```

➢ addKeyupEventToHtmlElement

```
addKeyupEventToHtmlElement(element, event)
```

Funzionamento:

La funzione aggiunge l’evento *keyup* (su tutti i tasti della tastiera) all’oggetto DOM contenuto in **element**. In **event** deve essere passato la funzione da associare al *keyup*

Esempio:

HTML:

```
<div id= “name”> Boolean </div>
```

Python:

```
def print_func():
      writeToConsole(“done!”)

element = getHtmlElement(“name”)
addKeyupEventToHtmlElement (element, print_func)
```

Console (alla pressione di un tasto):

```
>> done!
```

➢ checkIfEventIsEnterKey

```
checkIfEventIsEnterKey (event)
```

Funzionamento:

Restituisce vero se l'evento contenuto in **event** è scatenato dalla pressione del tasto "Enter". Altrimenti restituisce falso.

Esempio:

HTML:

```
<input id= "name"> </input>
```

Python:

```
def print_func(event):
      if(checkIfEventIsEnterKey(event))
            writeToConsole("done!")

element = getHtmlElement("name")
addKeyupEventToHtmlElement (element, print_func())
```

Console (alla pressione di enter):

```
>> done!
```