Python orienté web

Introduction à Brython

Par Deusyss

Date de publication : 12 novembre 2017

DÉBUTANT

Lorsque qu'un développeur Python souhaite commencer à apprendre à développer un site web, il lui ai aisé de trouver un framework : bottle, flask, django, …

Chacun de ces frameworks lui permettra de réaliser du web avec son langage préféré. Cependant, si cela lui permettra de réaliser du code dit " backend ", ou serveur, il lui faudra apprendre des langages complémentaires.

Orienté " frontend ", ou client léger, le HTML, le CSS, et le javascript sont les langages à connaître. Si le HTML et le CSS sont incontournables, une alternative Python existe cependant au dernier : Brython.

C'est ce framework que je vous invite ici à découvrir.

I - Introduction....4
II - Installation....4
III - Les bases....4
III-A - Les mots clés....4
III-B - Les fonctions intégrées de base....5
III-C - import de code externe....5
III-D - Intégration aux pages web....5
III-E - Optimisation post développement....6
III-F - Compilation....6
III-G - Hello world....7
III-H - Fonctionnalités basiques....7
III-H-1 - Le paquet browser....7
III-H-2 - Afficher une pop-up d'alerte....7
III-H-3 - Afficher une fenêtre d'impression....7
IV - Les événements....7
IV-A - Se connecter / se déconnecter à un événement....8
IV-A-1 - Attributs....8
IV-A-2 - Code exemple....8
IV-B - Souris....9
IV-B-1 - Événement....9
IV-B-2 - Attributs....9
IV-B-3 - Code exemple....10
IV-C - Clavier....11
IV-C-1 - Événements....11
IV-C-2 - Attributs....11
IV-C-3 - Code exemple....12
IV-D - Focus....13
IV-D-1 - Événements....13
IV-D-2 - Code exemple....13
IV-E - Glisser/déposer....13
IV-E-1 - Événements....13
IV-E-2 - Attributs....14
IV-E-3 - Code exemple....16
V - Gestion des éléments HTML....16
V-A - Balises HTML gérées....16
V-A-1 - HTML4....16
V-A-2 - HTML5....17
V-B - Accéder à un élément....17
V-B-1 - Par son id....17
V-B-2 - Par sa balise....17
V-B-3 - Par son nom....17
V-B-4 - Par son sélecteur CSS....18
V-C - Insertion d'un élément....18
V-C-1 - A la suite....18
V-C-2 - Avant un autre élément....18
V-C-3 - Après un autre élément....18
V-C-4 - Un tableau HTML....18
V-C-5 - Insérer un menu déroulant....19
V-C-6 - Insérer une image....19
V-D - Modifier un élément....19
V-D-1 - Modifier le texte....19
V-D-2 - Modifier le style....19
V-D-3 - Modifier la classe....19
V-D-4 - Modifier la mise en forme HTML....19
V-D-5 - Ajouter un descendant à un élément....20
V-D-6 - Supprimer un élément....20
V-E - Opérations sur élément....20
V-E-1 - Cacher/afficher un élément....20

V-E-2 - Animer un élément....20
V-E-3 - Effectuer une rotation sur un élément....21
V-E-4 - Déplacer un élément avec une souris....21
V-E-5 - Itérer sur des enfants....21
V-E-6 - Récupérer les valeurs des champs d'un formulaire....21
V-F - Attributs et méthodes des éléments....22
V-F-1 - Génériques....22
V-G - Opérations complémentaires....23
V-G-1 - Dessiner....23
V-G-2 - Utiliser le local storage....23
V-G-3 - Effectuer une requête AJAX....24
V-G-4 - Parser un fichier XML....24
V-G-5 - Parser un fichier JSON....24
V-H - Interaction avec les objets javascripts....24
V-H-1 - Date....24
V-H-2 - REGEX et chaîne de caractères....25
VI - Conclusion....25
VII - Remerciements....25

I - Introduction

Créer en 2014, **Brython** est, pour résumer grossièrement, un interpréteur Python, codé en javascript.

Le but est de pouvoir remplacer aisément le javascript de nos pages web par du code Python 3.

Bien entendu, en tant qu'interprété, il ne saura être aussi rapide que du code natif. Mais il existe nombre de cas ou il sera suffisant.

Comme tout projet récent, il possède encore quelques soucis de jeunesse. Néanmoins, vous aurez l'occasion, dans cet article, de constater qu'il s'agit d'un projet prometteur, et qu'il saura répondre à de nombreux besoins.

Que vous désiriez vous mettre au web sans apprendre javascript, ou simplement curieux, je vous invite à commencer cet article.

Cet article est écrit pour Python3.x. Des différences peuvent exister avec Python2.x.

Cet article fait référence à des notions HTML et CSS. Il est donc recommandé que les lecteurs disposent déjà de quelques notions sur ces langages.

L'intégralité de vos fichiers doivent être encodés en UTF8

II - Installation

Brython est disponible sur pypi. Aussi, l'installation sera on ne peux plus simple.

```
pip install brython
```

A partir de là, si vous allez voir l'installation effectuée, vous aurez alors les trois fichiers suivants :

- **brython.js** : L'interpréteur brython même, à inclure dans le code HTML
- **brython_stdlib.js** : Contient toute la librairie standard pour l'interpréteur
- **demo.html** : Une page de démonstration avec quelques exemples intégrés

III - Les bases

III-A - Les mots clés

Les mots clés Python suivants, sont les seuls à être gérés pour le moment :

- as
- assert
- break
- classe
- continue
- def
- del
- elif
- False
- finally
- for
- from
- global
- if
- import
- is
- nonlocal
- pass
- return
- True
- try
- while
- with
- yield

- else
- except
- lambda
- None

III-B - Les fonctions intégrées de base

Les fonctions intégrées dans Brython, de base, sont les suivantes :

- abs()
- all()
- any()
- ascii()
- bin()
- bool()
- bytes()
- callable()
- chr()
- classmethod()
- delattr()
- dict()
- dir()
- divmod()
- enumerate()
- eval()
- exec()
- filter()
- float()
- frozenset()
- getattr()
- globals()
- hasattr()
- hash()
- hex()
- id()
- input()
- int()
- isinstance()
- iter()
- len()
- list()
- locals()
- map()
- max()
- min()
- next()
- object()
- open()
- ord()
- pow()
- print()
- property()
- range()
- repr()
- reversed()
- round()
- set()
- setattr()
- slice()
- sorted()
- str()
- sum()
- super()
- tuple()
- type()
- zip()
- __import_()

Voici quelques précisions sur certaines de ces fonctions :

- **del()** : contrairement à du Python pur, cette fonction n'est pas automatiquement appelée lorsqu'un objet n'est plus utilisé. Il faut donc l'appeler manuellement.
- **open()** : cette fonction prend comme argument l'url du fichier à ouvrir. Ce fichier doit être contenu dans le même domaine. Les méthodes liées disponibles sont " read ", " readlines ", " seek ", " tell ", " close ".
- **print()** : par défaut la sortie s'effectue dans la console web

III-C - import de code externe

Dans n'importe lequel de vos scripts Brython, vous avez la possibilité d'importer du code externe Python, à condition que ce dernier soir positionné à la racine de votre projet web.

Dans vos scripts Brython, lorsqu'un import est effectué, Brython recherche le module/package demandé dans l'ordre suivant :

- *libs pour le code javascript et Lib pour Python*
- *Lib/site-packages dans Python*
- *Le dossier de la page courante*

III-D - Intégration aux pages web

Afin de pouvoir utiliser Brython dans votre code HTML, il vous faudra agir à deux niveaux :

- Spécifier qu'il faut utiliser Brython au lieu de javascript
- Spécifier le fichier contenant votre code Brython

Voici un aperçu type d'un fichier HTML :

```
1. <html>
2.     <head>
3.         <script type="text/javascript" src="brython.js"></script>
4.         <script type="text/javascript" src="brython_stdlib.js"></script>
5.     </head>
6.     <body onload="brython()">
7.         <script type="text/python" src="test.py"></script>
8.         <input id="zone"><button id="mybutton">click!</button>
9.      </body>
10. </html>
```

Les chemins des fichiers sont ici relatif.

Ligne 3, nous retrouvons l'import de l'interprêteur Brython. Une fois ce dernier importé, il faut l'activer. C'est la raison d'être de l'appel à la fonction " brython() " dans la balise " body ", ligne 5. Enfin, ligne 6, nous stipulons dans quel fichier trouver notre code Brython.

il est possible de passer des options lors de l'appel à la fonction " brython() ".

Les options possibles sont les suivantes :

- *0 : valeur par défaut. Aucun débogage activé*
- *1 : Les messages d'erreur sont imprimés dans la console du navigateur*
- *2 :La traduction du code Python en code javascript est affichée dans la console*
- *10 : La traduction du code Python et des modules importés est affichée dans la console*

III-E - Optimisation post développement

Afin d'optimiser les performances de vos développements Brython, l'outil vous offre la possibilité de générer un fichier " brython_modules.js " qui ne contiendra que le strict nécessaire pour votre projet.

Pour cela, depuis une console, placez vous dans le dossier racine de votre projet, puis exécutez la commande suivante :

```
python -m brython -modules
```

Il ne vous reste alors plus qu'à placer le fichier généré à la racine de votre projet, puis à remplacer, dans vos fichiers HTML, la ligne

```
<script type="text/javascript" src="brython_stdlib.js"></script>
```

par la ligne

```
<script type="text/javascript" src="brython_modules.js"></script>
```

III-F - Compilation

Afin de disposer directement su code javascript, il faut appeler la fonction 10 dans l'appel de " Brython(10) ".

Ce faisant, le code javascript équivalent sera écrit dans la console. Il ne vous restera alors plus qu'à le copier/coller dans un fichier et à mettre à jour vos pages web.

III-G - Hello world

Pour ceux qui ne le saurait pas, " document " est un terme désignant le contenu d'une page web.

La première chose à faire est donc d'importer le nécessaire donnant accès au document.

```
1. from browser import document
2. document <= "Hello world !"
```


Ici l'opérateur " <= " ne signifie pas " inférieur ou égal " mais " ajouter le contenu suivant ".

III-H - Fonctionnalités basiques

III-H-1 - Le paquet browser

Le paquet " browser " définit les noms et modules intégrés à Brython. Voici les principales méthodes. Nous en verrons d'autres par la suite.

Méthode	Description
browser.alert(<message>)	Affiche le message dans une pop-up
browser.confirm(<message>)	Affiche le message dans une pop-up. La pop-up possède deux boutons : OK et ANNULER. Retourne True si OK, False sinon.
browser.document	Renvoie un objet représentant le document HTML
browser.prompt(<message>)	Affiche le message dans une pop-up. La pop-up possède une zone de saisie.
browser.window	Renvoie un objet représentant la fenêtre du navigateur.

III-H-2 - Afficher une pop-up d'alerte

```
1. from browser import alert
2. alert("Hello !")
```

III-H-3 - Afficher une fenêtre d'impression

Il suffit d'appeler la méthode print de window

```
1. from browser import window
2. # ...
3. window.print(text)
```

IV - Les événements

Nous allons voir ici les divers événements gérés par Brython.

IV-A - Se connecter / se déconnecter à un événement

Vous avez le choix d'écouter ou non les divers événements, via les mots clés suivant:

- bind
- unbind

Le premier argument doit être le type d'événement auquel il doit réagir. Le deuxième est la fonction à appeler. Cette fonction ne devra prendre en compte qu'un seul argument : l'évènement qui l'appel, souvent abrégé en " ev ".

IV-A-1 - Attributs

Voici les attributs de " ev " :

Attribut	Description
bubbles	un booléen qui indique si l'élément se propage aux parents de l'élément sur lequel l'événement s'est produit
cancelable	un booléen qui indique si on peut annuler l'événement
currentTarget	l'élément sur lequel on est en train de traiter l'événement
defaultPrevented	booléen qui indique si on a appelé la méthode preventDefault() sur l'élément
eventPhase	indique quelle phase du flux d'événement est en cours de traitement
target	l'élément sur lequel l'événement s'est produit
timeStamp	la date/heure à laquelle l'événement s'est produit (en millisecondes depuis le 1/1/1970 à 0h)
type	le type d'événement

Par exemple, pour accéder à l'élément sur lequel l'évènement s'est produit, il vous faudra saisir " ev.target ".

IV-A-2 - Code exemple

```
1. from browser import document
2. from browser import alert
3.
4. def myevent(ev):
5.     alert('ça marche !')
6.
7. def compteur():
8.     alert('%s événement(s) attaché(s) à "click"'
9.         %len(document['myblock'].events('click')))
10.
11. def attache(ev):
12.     document['myblock'].bind('click', myevent)
13.     compteur()
14.     document['mymessage'].text='événement attaché, cliquer pour voir...'
15.
16. def detache(ev):
17.     if document['myblock'].events('click'):
18.         document['myblock'].unbind('click', myevent)
19.         compteur()
20.         document['mymessage'].text='clic désactivé'
21.
```

```
22. document['attache'].bind('click', attache)
23. document['detache'].bind('click', detache)
```

IV-B - Souris

IV-B-1 - Événement

Les événements disponibles pour le clavier sont les suivants :

Évenement	Description
mouseenter	la souris entre dans la zone couverte par l'élément, ou un de ses descendants
mouseleave	la souris sort de la zone couverte par l'élément et par ses descendants
mouseover	la souris entre dans la zone couverte par l'élément
mouseout	la souris quitte la zone couverte par l'élément
mousemove	la souris se déplace sur l'élément
mousedown	appui sur le bouton gauche de la souris
mouseup	relâchement du bouton gauche de la souris
click	clic : appui puis relâchement du bouton gauche de la souris
dblclick	double clic

Les événments " mouseenter " et " mouseleave " sont déclenchés quand la souris entre ou sort d'un élément. Si un élément en englobe d'autres, l'événement est déclenché à chaque fois que la souris entre ou sort d'un élément fils.

La différence avec mouseenter et mouseleave est qu'une fois que la souris est entrée dans un élément, l'événement n'est pas déclenché sur les éléments fils

IV-B-2 - Attributs

Attribut	Description
button	le numéro du bouton sur lequel on a appuyé
buttons	indique sur quels boutons de la souris on a appuyé pour déclencher l'événement. Chaque bouton sur lequel on peut appuyer est représenté par un entier donné: • 1 : bouton gauche • 2 : bouton droit • 4 : roue Si on appuie sur plus d'un bouton, la valeur de buttons est combinée pour produire un nouveau nombre. Par exemple, si on appuie sur le bouton droit

	(2) et sur la roue (4), la valeur est égale à 2|4, soit 6
x	la position de la souris par rapport au bord gauche de la fenêtre (en pixels)
y	la position de la souris par rapport au bord haut de la fenêtre (en pixels)
clientx	la position de la souris par rapport au bord gauche de l'élément dans lequel la souris se trouve au moment du clic (en pixels)
clienty	la position de la souris par rapport au bord haut de l'élément dans lequel la souris se trouve au moment du clic (en pixels)

IV-B-3 - Code exemple

mouseenter & mouseleave

```
1. from browser import document
2.
3. def _mouseenter(ev):
4.     document["trace1"].text = 'entrée dans %s' %ev.currentTarget.id
5.
6. def _mouseleave(ev):
7.     document["trace1"].text = 'sortie de %s' %ev.currentTarget.id
8.
9. document["jaune1"].bind('mouseenter',_mouseenter)
10. document["jaune1"].bind('mouseleave',_mouseleave)
11. document["bleu1"].bind('mouseenter',_mouseenter)
12. document["bleu1"].bind('mouseleave',_mouseleave)
```

mouseover & mouseout

```
1. from browser import document
2.
3. def _mouseover(ev):
4.     document["trace2"].text = 'entrée dans %s' %ev.currentTarget.id
5.
6. def _mouseout(ev):
7.     document["trace2"].text = 'sortie de %s' %ev.currentTarget.id
8.
9. document["jaune2"].bind('mouseover',_mouseover)
10. document["jaune2"].bind('mouseout',_mouseout)
11. document["bleu2"].bind('mouseover',_mouseover)
12. document["bleu2"].bind('mouseout',_mouseout)
```

mousemove

```
1. from browser import document
2.
3. def _mousemove(ev):
4.     document["trace3"].text = 'coordonnées : %s, %s' %(ev.x,ev.y)
5.
6. document["vert"].bind('mousemove',_mousemove)
```

IV-C - Clavier

IV-C-1 - Événements

Évenement	Description
input	déclenché quand la valeur d'un élément <input> ou <textarea> est modifié
keydown	appui sur une touche quelconque du clavier
keypress	appui sur une touche du clavier qui produit un caractère. Par exemple, quand on entre Ctrl+C au clavier, l'événement keypress n'est déclenché qu'au moment où on appuie sur C, alors que keydown est déclenché dès l'appui sur Ctrl
keyup	relâchement d'une touche enfoncée

IV-C-2 - Attributs

Attribut	Description
altKey	booléen, indique si la touche Alt (ou Option sur Mac) était enfoncée quand l'événement clavier a été déclenché Cet attribut n'est pas disponible pour l'événement input. Il est normalement utilisé avec keypress, pour pouvoir tester si on a entré Alt+<key> ou seulement <key>
charCode	Le numéro de référence Unicode pour la touche. Cet attribut n'est utilisable que pour l'événement keypress
ctrlKey	booléen, indique si la touche Ctrl était enfoncée quand l'événement clavier a été déclenché. Cet attribut n'est pas disponible pour l'événement input. Il est normalement utilisé avec keypress, pour pouvoir tester si on a entré Ctrl+<key> ou seulement <key>
keyCode	un code numérique dépendant du système et de l'implémentation, caractérise la clé enfoncée cette valeur est la même que les touches Alt, Ctrl ou majuscules soient enfoncées ou non. Notez que le résultat n'est pas le même selon qu'on gère les événements keydown, keyup et keypress
shiftKey	booléen, indique si la touche Majuscule était enfoncée quand l'événement clavier a été déclenché Cet attribut n'est pas disponible pour l'événement input. Il est normalement utilisé avec keypress, pour pouvoir tester

	si on a entré Shift+<key> ou seulement <key>
which	un code numérique dépendant du système et de l'implémentation, caractérise la clé enfoncée Notez que le résultat n'est pas le même selon qu'on gère les événements keydown, keyup et keypress

IV-C-3 - Code exemple

altKey
```
1. from browser import document as doc
2.
3. def altKey(ev):
4.     doc["traceAltKey"].text = 'altKey : %s ' %ev.altKey
5.
6. # le champ de saisie a comme id "altKey"
7. doc['altKey'].bind('keypress', altKey)
```

charCode
```
1. from browser import document as doc
2.
3. def charCode(ev):
4.     trace = doc["traceCharCode"]
5.     char = chr(ev.charCode)
6.     trace.text = 'charCode : %s, ' %ev.charCode
7.     trace.text += 'character : %s' %char
8.
9. doc['charCode'].bind('keypress', charCode)
```

ctrlKey
```
1. from browser import document as doc
2.
3.
4. def ctrlKey(ev):
5.     doc["traceCtrlKey"].text = 'ctrlKey : %s ' %ev.ctrlKey
6.     ev.preventDefault()
7.
8. doc['ctrlKey'].bind('keypress', ctrlKey)
```

Notez que ev.preventDefault() est appelé pour éviter le comportement par défaut associé à certains raccourcis clavier qui utilisent la touche Ctrl

keyCode
```
1. from browser import document as doc
2.
3. def keyCode(ev):
4.     trace = doc["traceKeyCode"]
5.     trace.text = 'event %s '%ev.type
6.     trace.text += ', keyCode : %s ' %ev.keyCode
7.     ev.stopPropagation()
8.
9. doc['keyCodeKeydown'].bind('keydown', keyCode)
10. doc['keyCodeKeypress'].bind('keypress', keyCode)
11. doc['keyCodeKeyup'].bind('keyup', keyCode)
```

shiftKey
```
1. from browser import document as doc
2.
3. def shiftKey(ev):
4.     doc["traceShiftKey"].text = 'shiftKey : %s ' %ev.shiftKey
```

shiftKey
```
5. doc['shiftKey'].bind('keypress', shiftKey)
```

which
```
1. from browser import document as doc
2.
3. trace = doc["traceWhich"]
4. def which(ev):
5.     trace.html = 'event : %s<br>' %ev.type
6.     trace.html += 'which : %s<br>' %ev.which
7.     if ev.type == 'keypress':
8.         trace.html += 'character : %s' %chr(ev.which)
9. doc['whichKeydown'].bind('keydown', which)
10. doc['whichKeypress'].bind('keypress', which)
11. doc['whichKeyup'].bind('keyup', which)
```

IV-D - Focus

IV-D-1 - Événements

Évenement	Description
blur	un élément a perdu le focus
focus	un élément a reçu le focus

IV-D-2 - Code exemple

```
1. from browser import document
2.
3. def getFocus(ev):
4.     document["traceFocus"].text = '%s reçoit le focus' %ev.target.id
5.
6. def loseFocus(ev):
7.     document["traceFocus"].text = '%s perd le focus' %ev.target.id
8.
9. document['entry'].bind('blur', loseFocus)
10. document['entry'].bind('focus', getFocus)
```

IV-E - Glisser/déposer

IV-E-1 - Événements

Évenement	Description
drag	Cet évènement est déclenché à la source du glisser-déposer, l'élément sur lequel l'évènement dragstart a été déclenché
dragend	La source du glisser-déposer recevra un évènement de ce type lorsque l'opération de glisser-déposer est terminée, qu'elle se soit bien déroulée ou non
dragenter	Déclenché lorsque le pointeur de la souris est déplacée pour la première fois au dessus d'un élément pendant le glisser-déposer. Un écouteur d'évènement pourrait alors indiquer si le dépôt des données courantes est autorisée ou non sur cette zone. Si aucun écouteur n'a été défini, ou si ce

	dernier n'entraîne aucune action, alors, le dépôt n'est par défaut, pas autorisé. C'est également l'évènement à prendre en charge pour donner des retours à l'utilisateur quand à la possibilité qu'il a déposer le contenu du glisser-déposer en affichant une surbrillance ou un marqueur d'insertion
dragleave	Cet évènement est déclenché quand la souris quite un élément durant un glisser-déposer. Les écouteurs évènement devraient retirer toute surbrillance ou marqueur d'insertion de cette zone
dragover	Cet évènement est déclenché lorsque la souris est déplacée au dessus d'un élément durant un glisser-déposer. La plupart du temps, cet évènement est utilisé pour les mêmes buts que l'évènement dragenter
dragstart	Déclenché sur un élément lorsque qu'un glisser-déposer est entrepris. L'utilisateur requiert la possibilité de glisser-déposer l'élément sur lequel cet évènement est déclenché
drop	L'évènement drop est déclenché sur l'élément sur lequel le dépôt a été effectué à la fin de l'opération de glisser déposer. Un écouteur d'évènement devrait être responsable de la récupération des données sources du glisser-déposer et de leur insertion sur la zone de dépôt. Cet évènement ne sera déclenché que si le dépôt est désiré. Il ne sera pas déclenché si l'utilisateur annule ce dernier en pressant, par exemple, sur la touche "Echap" de son clavier ou si le bouton de la souris a été relâché alors que le curseur était au-dessus d'une zone pour laquelle le glisser-déposer n'était pas autorisé

IV-E-2 - Attributs

Attribut	Description
dataTransfer	un "magasin de données" utilisé pour transporter des informations pendant le processus de glisser-déposer

Attribut de dataTransfer	Description
dropEffect	Une chaine qui représente l'effect qui sera utilisé, il doit toujours être une des valeurs possibles de effectAllowed. Pour les événements dragenter et dragover, le dropEffect sera initialisé

	en fonction de l'action requise par l'utilisateur. La manière dont cela est déterminé dépend de la plateforme, mais typiquement l'utilisateur peut appuyer sur des touches de modification pour préciser l'action souhaitée. A l'intérieur d'un gestionnaire d'événement pour dragenter ou dragover, le dropEffect doit être modifié si l'action que l'utilisateur requiert n'est pas celle qui est souhaitée. Pour les événements dragstart, drag et dragleave, le dropEffect est initialisé à "none". On peut affecter une valeur à dropEffect, mais cette valeur ne sera pas utilisée. Pour les événements drop et dragend, la valeur de dropEffect sera l'action souhaitée, c'est-à-dire celle que dropEffect avait après le dernier événement dragenter ou dragover. Les valeurs possibles sont : • "copy" : une copie de l'élément source est effectuée dans le nouvel emplacement. • "move" : un élément est déplacé dans le nouvel emplacement. • "link" : un lien vers l'élément source est établie dans le nouvel emplacement. • "none" : l'élément ne peut pas être déposé. Affecter une autre valeur n'a aucun effet et ne modifie pas la valeur courante.
effectAllowed	Une chaine qui spécifie les effets autorisés pour le déplacement. On peut le définir dans l'événement dragstart pour indiquer les effets souhaités pour la source, et dans les événements dragenter et dragover pour insiquer les effets souhiatés pour la cible. La valeur n'est pas utilisée pour les autres événements. Les valeurs possibles sont: • "copy" : une copie de la source peut être effectuée au nouvel emplacement. • "move" : un élément peut être déplacé au nouvel emplacement. • "link" : un lien peut être établi vers la source dans le nouvel emplacement. • "copyLink" : une opération de copie ou de lien est autorisée.

	• "copyMove" : une opération de copie ou de déplacement est autorisée. • "linkMove" : une opération de lien ou de déplacement est autorisée. • "all" : toutes les opérations sont autorisées. • "none" : l'élement ne peut pas être déposé. • "uninitialized" : la valeur par défaut quand l'effet n'a pas été défini, équivalent à "all". Affecter une autre valeur n'a aucun effet et ne modifie pas la valeur courante.
files	La liste de tous les fichiers locaux disponibles pour le transfert de données. Si l'opération de déplacement n'implique pas de glisser-déposer de fichiers, cette propriété est une liste vide. Une tentative d'accès à un élément de cette liste avec un index non valide renvoie None.
getData(type)	Récupère la donnée pour un type donné, ou une chaine vide si la donnée pour ce type n'existe pas ou que l'attribut dataTransfer ne contient aucune donnée
setData(type, valeur)	Affecte une valeur à un type donné. S'il n'y a pas de données pour ce type, elle est ajoutée à la fin, de façon que le dernier élément de la liste des types sera le nouveau format. Si la donnée pour ce type existe déjà, la valeur courante est remplacée à la même position. Autrement dit, l'ordre de la liste des types n'est pas modifiée quand on change la valeur pour un type.
types	La liste des types de formats des données stockées pour le premier élément, dans l'ordre où les données ont été ajoutées. Si aucune donnée n'a été ajoutée, cette propriété est une liste vide.

IV-E-3 - Code exemple

V - Gestion des éléments HTML

V-A - Balises HTML gérées

V-A-1 - HTML4

- A
- ABBR
- ACRONYM
- ADDRESS
- DFN
- DIR
- DIV
- DL
- INPUT
- INS
- ISINDEX
- KBD
- SCRIPT
- SELECT
- SMALL
- SPAN

- APPLET
- AREA
- B
- BASE
- BASEFONT
- BDO
- BIG
- BLOCKQUOTE
- BODY
- BR
- BUTTON
- CAPTION
- CENTER
- CITE
- CODE
- COL
- COLGROUP
- DD
- DEL
- DT
- EM
- FIELDSET
- FONT
- FORM
- FRAME
- FRAMESET
- H1
- H2
- H3
- H4
- H5
- H6
- HEAD
- HR
- HTML
- I
- IFRAME
- IMG
- LABEL
- LEGEND
- LI
- LINK
- MAP
- MENU
- META
- NOFRAMES
- NOSCRIPT
- OBJECT
- OL
- OPTGROUP
- OPTION
- P
- PARAM
- PRE
- Q
- S
- SAMP
- STRIKE
- STRONG
- STYLE
- SUB
- SUP
- SVG
- TABLE
- TBODY
- TD
- TEXTAREA
- TFOOT
- TH
- THEAD
- TITLE
- TR
- TT
- U
- UL
- VAR

V-A-2 - HTML5

- ARTICLE
- ASIDE
- AUDIO
- BDI
- CANVAS
- COMMAND
- DATA
- DATALIST
- EMBED
- FIGCAPTION
- FIGURE
- FOOTER
- HEADER
- KEYGEN
- MAIN
- MARK
- MATH
- METER
- NAV
- OUTPUT
- PROGRESS
- RB
- RP
- RT
- RTC
- RUBY
- SECTION
- SOURCE
- SUMMARY
- TEMPLATE
- TIME
- TRACK
- VIDEO
- WBR
- DETAILS (5.1)
- DIALOG (5.1)
- MENUITEM (5.1)
- PICTURE (5.1)
- SUMMARY (5.1)

V-B - Accéder à un élément

V-B-1 - Par son id

```
1. from browser import document
2. data = document["data"]
```

V-B-2 - Par sa balise

```
1. from browser import html
2. # ...
3. links = document[html.A]
```

V-B-3 - Par son nom

```
1. from browser import html
2. # ...
3. tmp = elt.get(name= "Hello")
```

V-B-4 - Par son sélecteur CSS

```
from browser import html
# ...
tmp = elt.get(selector=S)
```

V-C - Insertion d'un élément

V-C-1 - A la suite

```
from browser import document, html
element = document["zone6"]
nb = 0
def change(event):
    global nb
    element <= html.B(" {}".format(nb))
    nb += 1
document["button6"].bind("click", change)
```

V-C-2 - Avant un autre élément

```
from browser import document, html
ul = document["zone61"].get(selector="ul")[0]
element = ul.get(selector="li")[0]
nb = 0
def change(event):
    global nb
    nb += 1
    ul.insertBefore(html.LI(f"element before {nb}"), element)
document["button61"].bind("click", change)
```

V-C-3 - Après un autre élément

```
from browser import document, html
ul = document["zone62"].get(selector="ul")[0]
element = ul.get(selector="li")[0]
nb = 0
def change(event):
    global nb
    nb += 1
    ul.insertBefore(html.LI(f"element after {nb}"), element.nextSibling)
document["button62"].bind("click", change)
```

V-C-4 - Un tableau HTML

```
from browser import document as doc
from browser.html import TABLE, TR, TH, TD
table = TABLE()
row = TR() # create a row
# add header cells
row <= TH("Pays")
row <= TH("Capitale")
table <= row # add the row to the table

# add a row
row = TR()
row <= TD("Russie")+TD("Moscou")
table <= row

# erase initial content
doc['zone'].clear()

```

```
# insert table in the element
doc['zone'] <= table
```

V-C-5 - Insérer un menu déroulant

```
from browser import document, alert, html
def show(event):
    dropdown = event.target
    num = dropdown.selectedIndex
    alert("Selected: {}".format(dropdown.options[num].value))
def insert_dropdown(event):
    document["zone12"] <= 'Your choice : '
    dropdown = html.SELECT(html.OPTION("Choice {}".format(i)) for i in range(5))
    dropdown.bind('change', show)
     document["zone12"] <= dropdown
document['button12'].bind('click', insert_dropdown)
```

V-C-6 - Insérer une image

```
from browser import document, html
logo = "https://www.python.org/static/community_logos/python-logo-master-v3-TM.png"
def insert_image(event):
    document["zone9"].clear()
    document["zone9"] <= html.IMG(src=logo, height=50)
document["button9"].bind("click", insert_image)
```

V-D - Modifier un élément

V-D-1 - Modifier le texte

```
from browser import document
# ...
document["zone1"].textContent = "New content"
```

V-D-2 - Modifier le style

```
from browser import document
element = document["zone2"]
style = element.style
color = style.color
style.color = "#cc8" if color == "blue" else "blue"
style.backgroundColor = "gray" if color == "blue" else "#aad"
style.fontWeight = "bold" if color == "blue" else "normal"
style.fontSize = "18px" if color == "blue" else "14px"
```

V-D-3 - Modifier la classe

```
from browser import document
element = document["zone_class"]
element.classList.add("down")
if "down" in element.classList:
    element.classList.remove("down")
    element.classList.add("up")
```

V-D-4 - Modifier la mise en forme HTML

```
from browser import document, html

document['zone'].clear()
```

```
4. document['zone'] <= html.H1("Introduction à Brython")
5. document['zone'] <= html.H4(html.I("Python dans le navigateur"))
6. document['zone'] <= html.B("Salut !")
```

V-D-5 - Ajouter un descendant à un élément

```
1. from browser import document, html
2. # ...
3. document['zone'] <= html.INPUT(Id="data")
```

V-D-6 - Supprimer un élément

```
1. zone = document['zone']
2. del zone
```

V-E - Opérations sur élément

V-E-1 - Cacher/afficher un élément

```
1. from browser import document
2. display = document["zone3"].style.display
3. if display == 'inline' :
4.     document["zone3"].style.display = "none"
```

Ici, nous pouvons constater que placer le style.display à "none" revient à le cacher.

V-E-2 - Animer un élément

```
1. from browser import document, window
2. moving = document["rot15"]
3. x = 0
4. dx = 3
5. run = None
6. def change(event):
7.     global run
8.     if run is None:
9.         # start animation
10.         animloop(1)
11.     else:
12.         # stop animation
13.         window.cancelAnimationFrame(run)
14.         run = None
15. def render():
16.     global x, dx
17.     moving.style.transform = "translate({}px,0)".format(x)
18.     x += dx
19.     if x > document["zone15"].offsetWidth-moving.offsetWidth:
20.         dx = -dx
21.         moving.html = "&#9668;" # left triangle
22.     elif x <= 0:
23.         dx = -dx
24.         moving.html = "&#9658;" # right triangle
25. def animloop(t):
26.     global run
27.     run = window.requestAnimationFrame(animloop)
28.     render()
29. document['button15'].bind('click', change)
```

V-E-3 - Effectuer une rotation sur un élément

```
1. from browser import document, html
2. moving = document["rot14"]
3. angle = 10
4. def change(event):
5.     global angle
6.     moving.style.transform = "rotate({}deg)".format(angle)
7.     angle += 10
8. document['button14'].bind('click', change)
```

V-E-4 - Déplacer un élément avec une souris

```
1. from browser import document
2.
3. class ElementMove:
4.
5. def __init__(self, moving):
6.     """Make "moving" element movable with the mouse"""
7.     self.moving = moving
8.     self.is_moving = False
9.     self.moving.bind("mousedown", self.start)
10.     self.moving.bind("mousemove", self.move)
11.     self.moving.bind("mouseup", self.stop)
12.     moving.style.cursor = "move"
13.
14. def start(self, event):
15.     """When user clicks on the moving element, set boolean is_moving
16.     to True and store mouse and moving element positions"""
17.     self.is_moving = True
18.     self.mouse_pos = [event.x, event.y]
19.     self.elt_pos = [self.moving.left, self.moving.top]
20.     # prevent default behaviour to avoid selecting the moving element
21.     event.preventDefault()
22.
23. def move(self, event):
24.     """User moves the mouse"""
25.     if not self.is_moving:
26.         return
27.
28.     # set new moving element coordinates
29.     self.moving.left = self.elt_pos[0] + event.x - self.mouse_pos[0]
30.     self.moving.top = self.elt_pos[1] + event.y - self.mouse_pos[1]
31.
32. def stop(self, event):
33.     """When user releases the mouse button, stop moving the element"""
34.     self.is_moving = False
35.
36. ElementMove(document["moving"])
```

V-E-5 - Itérer sur des enfants

```
1. for child in element:
2.     # ...
```

V-E-6 - Récupérer les valeurs des champs d'un formulaire

```
1. from browser import document, html
2. def show_values(event):
3.     input = document["input16"].value
4.     select = document["select16"]
5.     option = select.options[select.selectedIndex].value
6.     text = document["textarea16"].value
7.     document["zone16"].clear()
8.     document["zone16"] <= ("Value in INPUT field: {}".format(input),
```

```
9.          html.BR(), "Selected option: {}".format(option),
10.          html.BR(), "Value in TEXTAREA field: {}".format(text)
11.          )
12. document['button16'].bind('click', show_values)
```

V-F - Attributs et méthodes des éléments

Voici un tableau récapitulant les possibilité offerte par Brython, relatif aux propriétés et aux méthodes des éléments de la page web.

V-F-1 - Génériques

Nom	Type	Description	Lecture	Écriture
abs_left	Entier	Position de l'élément par rapport au bord gauche de l'écran	X	
abs_top	Entier	Position de l'élément par rapport au bord supérieur de l'écran	X	
children	Liste	Les éléments "descendants" de l'élément	X	
class_name	Chaîne	Le nom de la classe de l'élément (attribut class de la balise)		X
clear	Méthode	" elt.clear() " supprime tous les descendants de l'élément		
get	Méthode	Sélectionne des éléments		
height	Entier	Hauteur de l'élément en pixels		X
html	Chaîne	Le code HTML contenu dans l'élément		X
index	Méthode	" elt.index() " renvoie le rang (entier) de l'élément parmi les enfants de son parent		
inside	Méthode	"elt.inside(autre)" teste si " elt " est contenu		

		dans l'élément " autre "		
left	Entier	La position de l'élément par rapport au bord gauche du premier parent positionné		X
parent	Instance de DOMNode	L'élément parent de l'élément (None pour document)	X	
select	Méthode elt.s	elect(css_selec renvoie les éléments correspondant au sélecteur CSS spécifié		
text	Chaîne	Le texte contenu dans l'élément		X
top	Entier	La position de l'élément par rapport au bord supérieur du premier parent positionné		X
width	Entier	Largeur de l'élément en pixels		X

V-G - Opérations complémentaires

V-G-1 - Dessiner

```
1. from browser import document, html
2. import math
3. canvas = document["zone8"]
4. ctx = canvas.getContext("2d")
5. x = 20
6. def draw(event):
7.     global x
8.     ctx.beginPath()
9.     ctx.arc(x, 25, 15, 0, 2*math.pi)
10.     x += 15
11.     ctx.stroke()
12. document["button8"].bind("click", draw)
```

V-G-2 - Utiliser le local storage

```
1. from browser import alert
2. from browser.local_storage import storage
3. import json
4.
5. a = {'foo':1,1515:'Marignan'}
6.
7. storage["brython_test"] = json.dumps(a)
8.
9. b = json.loads(storage['brython_test'])
```

```
10. alert(b['foo'])
11. alert(b['1515'])
```

V-G-3 - Effectuer une requête AJAX

```
1. from browser import document, ajax
2.
3. url = "http://api.open-notify.org/iss-now.json"
4. msg = "Position of the International Space Station at {}: {}"
5.
6. def complete(request):
7.     import json
8.     import datetime
9.     data = json.loads(request.responseText)
10.     position = data['iss_position']
11.     ts = data['timestamp']
12.     now = datetime.datetime.fromtimestamp(ts)
13.     document["zone10"].text = msg.format(now, position)
14.
15. def change(event):
16.     req = ajax.ajax()
17.     req.open('GET', url, True)
18.     req.bind('complete', complete)
19.     document["zone10"].text = "waiting..."
20.     req.send()
21.
22. document['button10'].bind('click', change)
```

V-G-4 - Parser un fichier XML

```
1. from browser import document, window, html
2. parser = window.DOMParser.new()
3.
4. def show(node, container):
5.     if hasattr(node, "tagName"):
6.         container <= html.STRONG(node.tagName)
7.         ul = html.UL()
8.         container <= ul
9.         for child in node.childNodes:
10.             show(child, ul)
11.     elif node.text.strip():
12.         container <= html.LI(node.text)
13.
14. def show_xml():
15.     src = open("cd-catalog.xml").read()
16.     tree = parser.parseFromString(src, "application/xml")
17.
18. root = tree.firstChild
19.
20. show(root, document["zone18"])
21.
22. document["button18"].bind("click", lambda ev: show_xml())
```

V-G-5 - Parser un fichier JSON

Le format JSON étant natif en Python, puisqu'il correspond à un dictionnaire, je vous renvoie simplement vers un **article dédié**.

V-H - Interaction avec les objets javascripts

V-H-1 - Date

```
1. from browser import document, window
```

```
2.
3. def show_date(event):
4.     date = window.Date.new()
5.     document['zone11'].text = '{}-{:02}-{:02} at {:02}:{:02}:{:02}'.format(
6.         date.getFullYear(), date.getMonth()+1, date.getDate(),
7.         date.getHours(), date.getMinutes(), date.getSeconds())
8. document['button11'].bind('click', show_date)
```

V-H-2 - REGEX et chaîne de caractères

```
1. from browser import document, window
2. def change(event):
3.     s = window.String.new("abracadabra")
4.     document['zone17'].text = s.replace(window.RegExp.new("a", "g"), "i")
5. document['button17'].bind('click', change)
```

VI - Conclusion

Comme nous venons de le voir ensemble, Brython possède un vrai potentiel, et répond à des attentes réelles.

Toujours en cours de développement, et relativement jeune, il s'agit néanmoins d'un projet prometteur. Et s'il ne saura remplacer systématiquement javascript, sa facilité d'apprentissage, et de mise en œuvre, en font en tout cas, un outil idéal pour du prototypage rapide.

J'espère que cet article vous aura intéressé et attisé votre curiosité concernant ce framework.

VII - Remerciements