Learning Javascript while working with Brython

What it is Brython?

Brython derives from Browser Python

Brython wants to be a 100% compliant Python3 implementation made with Python and Javascript that runs in the browser

Some time ago the question was: Why should I use javascript?

Two answers:

- The first is that you have no choice
- The second is that javascript could be really good (*)

Douglas Crockford (http://www.crockford.com/index.html), 'Javascript, the good parts'
(*) or at least it wouldn't be so bad as many people believe

Really? ;-)

In [2]: HTML('<iframe src=https://www.destroyallsoftware.com/talks/wat width=900 height=500></iframe>

Out[2]:

**Destroy All Software Talks**


**Wat**

A lightning talk by Gary Bernhardt from CodeMash 2012

**But it was some time ago. Nowadays people could**
In [3]: html = HTML('Iframe src=http://altjs.org/ width=900 height=350></iframe>

Out[3]:

In [4]: from IPython.display import HTML
   :

Out[4]:

<table>
<thead>
<tr>
<th>Project name</th>
<th>Gist</th>
<th>Actively developed?</th>
<th>Can import?</th>
<th>Compatibility with CPython 2 or 3</th>
<th>Python stdlib?</th>
<th>Can call Javascript code?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Python 3 translator</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Credits: Stromberg (http://stromberg.dnsalias.org/~strombrg/pybrowser/python-browser.html)

Other similar lists (https://github.com/jashkenas/coffee-script/wiki/List-of-languages-that-compile-to-JS).
WTF?? Why are you trying to do that?

Even GvR doesn't recommend to do that!!

Python in the browser ?

by Btrot69

Over the years, there have been several attempts to create a sandboxed version of python that will safely run in a web browser. Mostly this was because of problems with Javascript. Now that Javascript works -- and we have nice things like CoffeeScript -- is it time to give up on python in the browser?

Guido: I gave up on it in 1995, so yes. And please don’t try to compile Python to JavaScript. The semantics are so different that you end up writing most of a Python runtime in JavaScript, which slows things down too much. (CoffeScript’s strength is that it is designed to map cleanly to JavaScript, and the two are now co-evolving to make the mapping even cleaner.)


Brython, the good parts

Brython is a (subset of) CPython3 that runs in the browser

Python syntax

```python
a = 1
b = 2
print(a + b)

a, b = 1, 2
print(a + b)

print(type(a))
print(type(a * 1.0))

print(1 + 'a')
```
Functions

```python
def sum(a, b):
    return a + b
print(sum(2, 2))
```

Classes
class Square:
    def __init__(self, length):
        self.len = length

    def area(self):
        return self.len ** 2

sq = Square(2)
print(sq.area())

In [7]: HTML('<iframe src=http://brython.info/tests/console.html width=900 height=350></iframe>

Out[7]:

Inheritance
How classes are implemented in Brython

Python classes (including built-in classes) are implemented with 2 different Javascript objects: a factory function that creates instances (it uses __new__ and __init__ when available) and a dictionary that holds the class attributes and methods.

(*) super is not implemented in Brython 1.2 (well, it is since some days ago (https://bitbucket.org/olemis/brython/src/a062b0a69cd064bf0fd4131fc9323ee47416e5545c/src/py_builtin_functions.js?at=default#cl-1147)).
def world(some_func):
    def pre(arg):
        greet = some_func(arg)
        return greet + ' world!'
    return pre

@world
def whatever(word):
    return word

print(whatever('hello'))

Brython supports most keywords and functions of Python 3:

**Keywords**

as, assert, break, class, continue, def, del, elif, else, except, False, finally, for, from, global, if, import, is, lambda, None, pass, return, True, try, while, with, yield

**Built-in functions**
The following are not implemented in the current version:

**keywords**

- nonlocal

**built-in functions**

- bytearray(), compile(), complex(), format(), help(), memoryview(), super(), vars(), __import__

The complex number type (j) is not supported

Ok, I have Python to simulate Python :-) 

Show me the money!!!

Ok, let's see what Brython can do in the browser

First of all, to use Brython you need to:
• include the following in your html file

```html
<script type="text/javascript" src="path/to/the/library/brython.js">
</script>
```

• Include the following in the body tag

```html
<body onload="brython()">
</body>
```

• Include your python code in a script tag using text/python or text/python3

```html
<script type="text/python">
...Your Python code...
</script>
```

There are some things that are different to Python

By default, `print()` will output to the web browser console and so are the error messages. `sys.stderr` and `sys.stdout` can be assigned to an object with a `write()` method, and this allows for the redirection of output to go to a window or text area, for example.

`sys.stdin` is not implemented at this time, however there is an `input()` built-in function that will open a blocking input dialog (a prompt).

To open a print dialog (to a printer), call `win.print()`.

Some keywords and built-in functions designed for operation in a browser have been added:

- built-ins
  - `alert()`, `confirm()`, `prompt()`

  correspond to their Javascript equivalents

  the `win` keyword is the window (window object in JS) and `doc` represents the HTML document (document in JS).
In [10]: HTML('<iframe src=http://curious-electric.com/brython-playground/ width=900 height=400></iframe>

Out[10]:

<table>
<thead>
<tr>
<th>HTML</th>
<th>CSS</th>
<th>Python</th>
</tr>
</thead>
</table>
| <p>Your name? <input id="mynname"> <button onclick="echo()">click!</button> | body { color: blue; } | def echo():
  alert("Hello %s !" % doc["mynname"].value) |

Credits of the Brython jsFiddle clone (https://github.com/dirkk0/brython-playground): Dirk Krause.
How Can I access the HTML elements

Getting access to an element can be done in different ways. The most usual is to use its identifier, ie its attribute id : with an input field defined by

```html
<input id="data">
```

we can get a reference to this field by

```python
data = doc["data"]
```

doc is a built-in Brython keyword that refers to the HTML document. It behaves like a dictionary whose keys are the identifiers of the elements in the page. If not element has the specified id, the program raises a KeyError exception

We can also get all the elements of a given type, for instance all the hypertext links (HTML tag A), using the syntax

```python
import html
links = doc[html.A]
```

Finally, all the elements in the page have a method get() that can be used to search elements :

```python
elt.get(name=N) returns a list of all the elements descending from elt whose attribute name is equal to N
elt.get(selector=S) returns a list with all the elements descending from elt whose CSS selector matches S
```

Ok, let's see Brython in Action with some examples I created for the PyConES'2013!!
In [6]: HTML('<iframe src=https://googledrive.com/host/0B4OEtv-kAaTBSIIJM19hdkpCeTQ/colors_bootstrapped.html width=350></iframe>')</div>

Out[6]:

<table>
<thead>
<tr>
<th>Change div color (onmousedown example)</th>
</tr>
</thead>
<tbody>
<tr>
<td>This simple app only works in modern browsers</td>
</tr>
</tbody>
</table>

Link to colors app (https://googledrive.com/host/0B4OEtv-kAaTBSIIJM19hdkpCeTQ/colors_bootstrapped.html).

(Warning: to make this work you should run a server first so it is not working in the presentation)

In [9]: HTML('<iframe src=https://googledrive.com/host/0B4OEtv-kAaTBSIIJM19hdkpCeTQ/table_bootstrapped.html width=350></iframe>')</div>

Out[9]:

<table>
<thead>
<tr>
<th>HTML (+ Brython): Managing HTML</th>
</tr>
</thead>
<tbody>
<tr>
<td>Creating html using Brython.</td>
</tr>
</tbody>
</table>

Link to table app (https://googledrive.com/host/0B4OEtv-kAaTBSIIJM19hdkpCeTQ)
HTML5 Drag&Drop example made with Brython

This simple app has been tested only in Firefox and it's been adapted from this tutorial (http://dev.tutsplus.com/tutorials/create-an-html5-tile-swapping-puzzle--active-10747).

JSONP call example made with Brython

This simple app has been tested only in Firefox.
Link to jsonp request app (https://googledrive.com/host/0B4OEtv-kAaTBSllJM19hdkpCeTQ/jsonp_bootstrapped.html).

In [13]: HTML('<iframe src=https://googledrive.com/host/0B4OEtv-kAaTBSllJM19hdkpCeTQ/hangman_bootstrapped.html width=500></iframe>')</n

Out[13]:

Hangman game made with Brython

This simple app has been tested only in Firefox and it's been inspired by Jennifer Dewalt day 78 exercise (http://jenniferdewalt.com/game).

Link to hangman game app (https://googledrive.com/host/0B4OEtv-kAaTBSllJM19hdkpCeTQ/hangman_bootstrapped.html).
HTML5 (+ Brython): TO DO Application
Using Local Storage

This simple app only works in modern browsers.

Task

Text input

Relevance

High

Add

<table>
<thead>
<tr>
<th>Task</th>
<th>Relevance</th>
<th>Created</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Link to TO-DO list app (https://googledrive.com/host/0B4OEtv-kAaTBSIIJM19hdkpCeTQ/todo_bootstrapped.html).

https://github.com/kikocorreoso/Brython-PyConES-2013
(https://github.com/kikocorreoso/Brython-PyConES-2013)

@pybonacci (https://twitter.com/Pybonacci)

Official repository https://bitbucket.org/olemis/brython/overview
(https://bitbucket.org/olemis/brython/overview)

Official web page (http://brython.info/)

Apps running in my Gdrive (https://drive.google.com)
And that's all!

Now you should love Brython :-)


Out[3]:

![SpongeBob SquarePants](http://images.fanpop.com/images/image_uploads/sponge-bob-rocks-spongebob-squarepants-154588_427_600.jpg)