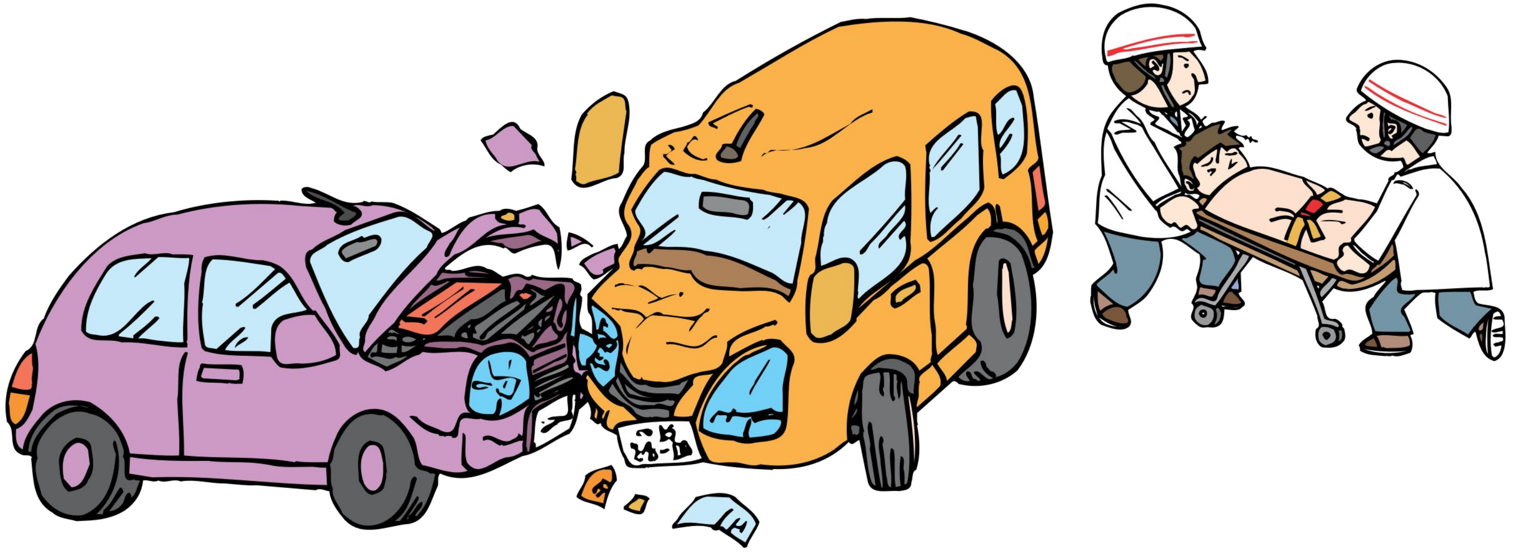
Big forms with JSON schemas and Transcrypt

November 15th, 2018

Philippe Entzmann

Reinsurance of car insurers



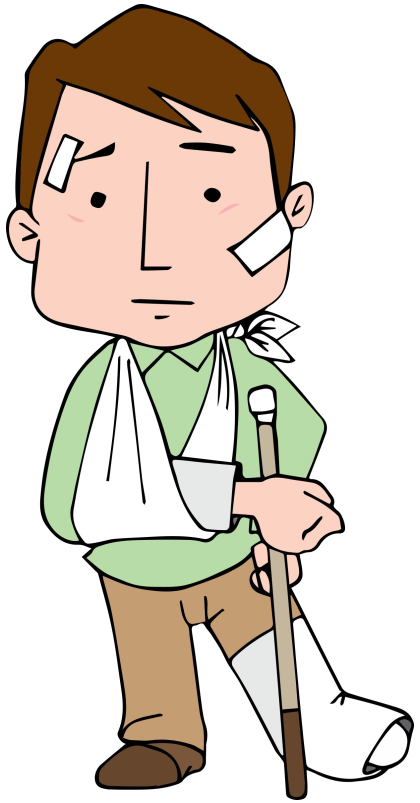
Victim’s injuries follow-up

Yearly evaluation over lifespan.

Detailed expenses tracking of physical and non-physical injuries to the victim and its relatives.

Reference to mortality tables and currency rate.

A lot of differently structured data to collect.



From written forms to a database

Written big forms from different sources and different shapes consolidated in a single database.

The data schema will highly evolve over time. Our experts have to manage the data schema themselves :

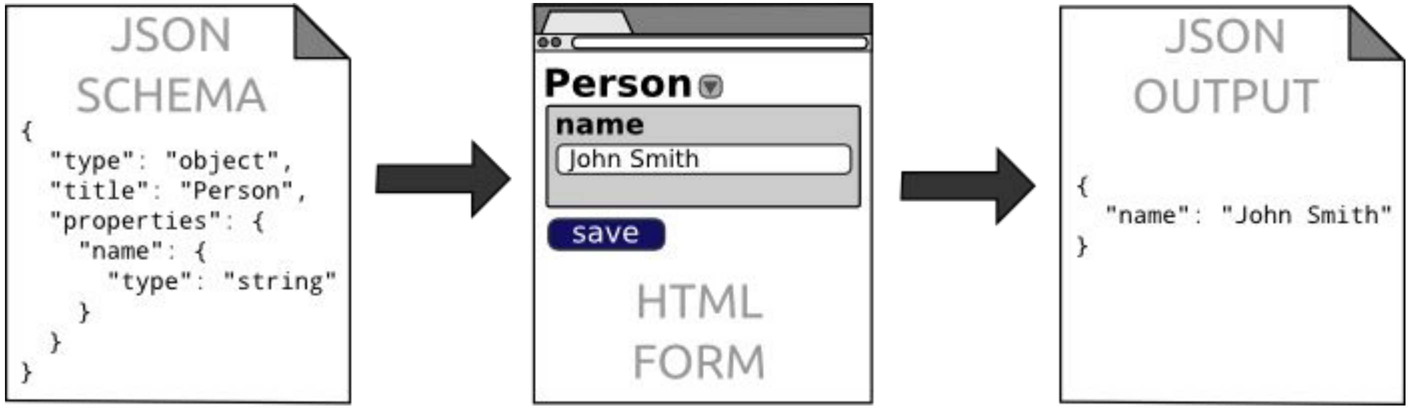
- add fields, nested fields, list, set properties, ...
- split the whole schema in reusable parts
- define simple but usefull formulas

Each form may use 30 reusable sub-form parts leading to 300 base fields per form for a filled document of more than 1000 fields.

From schema to web form

We choose the excellent json-editor library :

“JSON Editor takes a JSON Schema and uses it to generate an HTML form.”

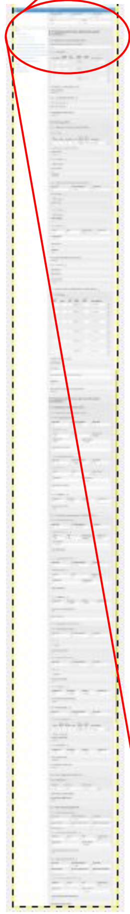


```
{
  "type": "object",
  "title": "Person",
  "properties": {
    "name": {
      "type": "string"
    }
  }
}
```

```
{
  "name": "John Smith"
}
```

Write the business rules in javascript

It worked



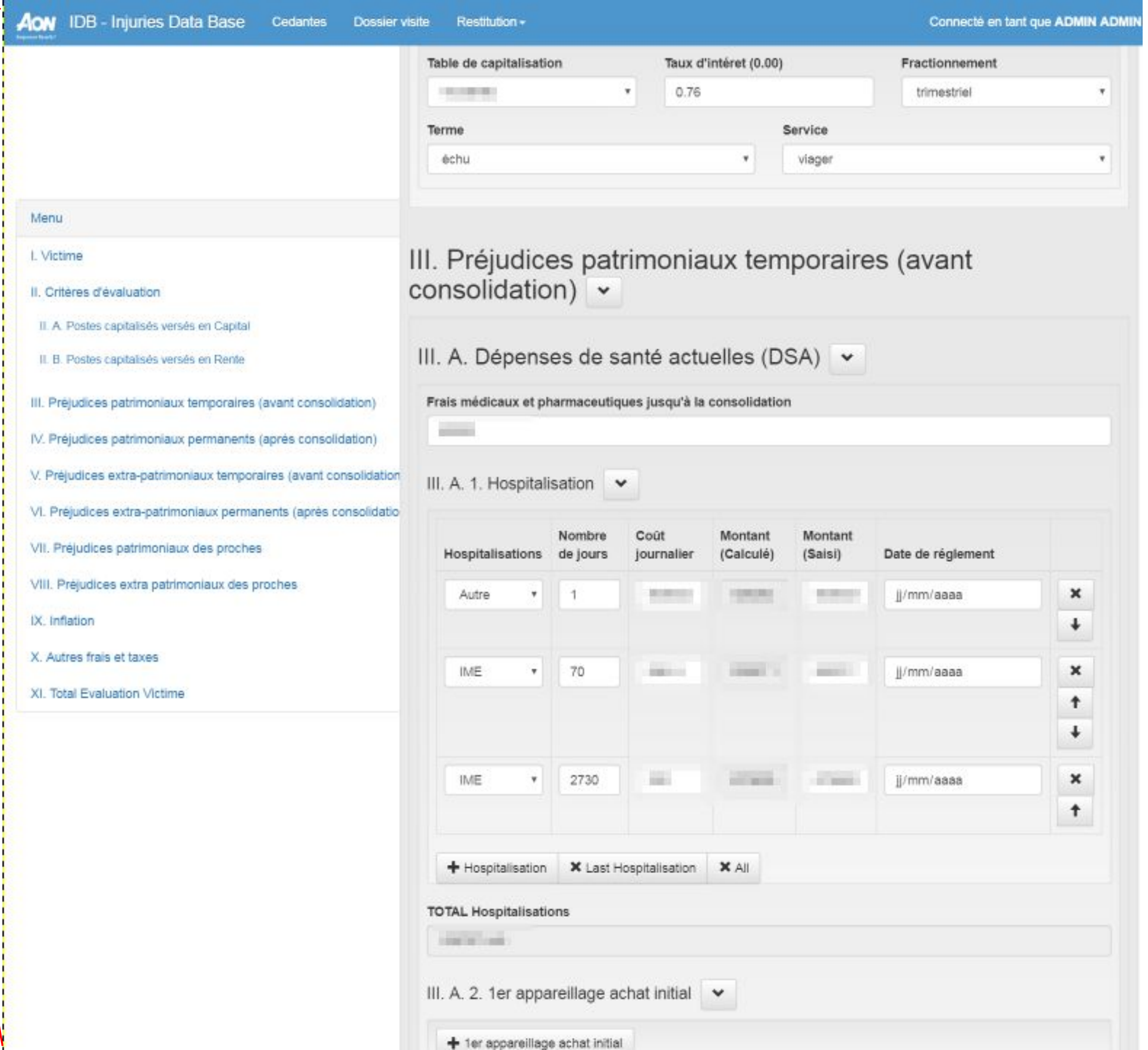


Table de capitalisation	Taux d'intérêt (0.00)	Fractionnement
[redacted]	0.76	trimestriel

Terme	Service
échu	viager

III. Préjudices patrimoniaux temporaires (avant consolidation)

III. A. Dépenses de santé actuelles (DSA)

Frais médicaux et pharmaceutiques jusqu'à la consolidation

[redacted]

III. A. 1. Hospitalisation

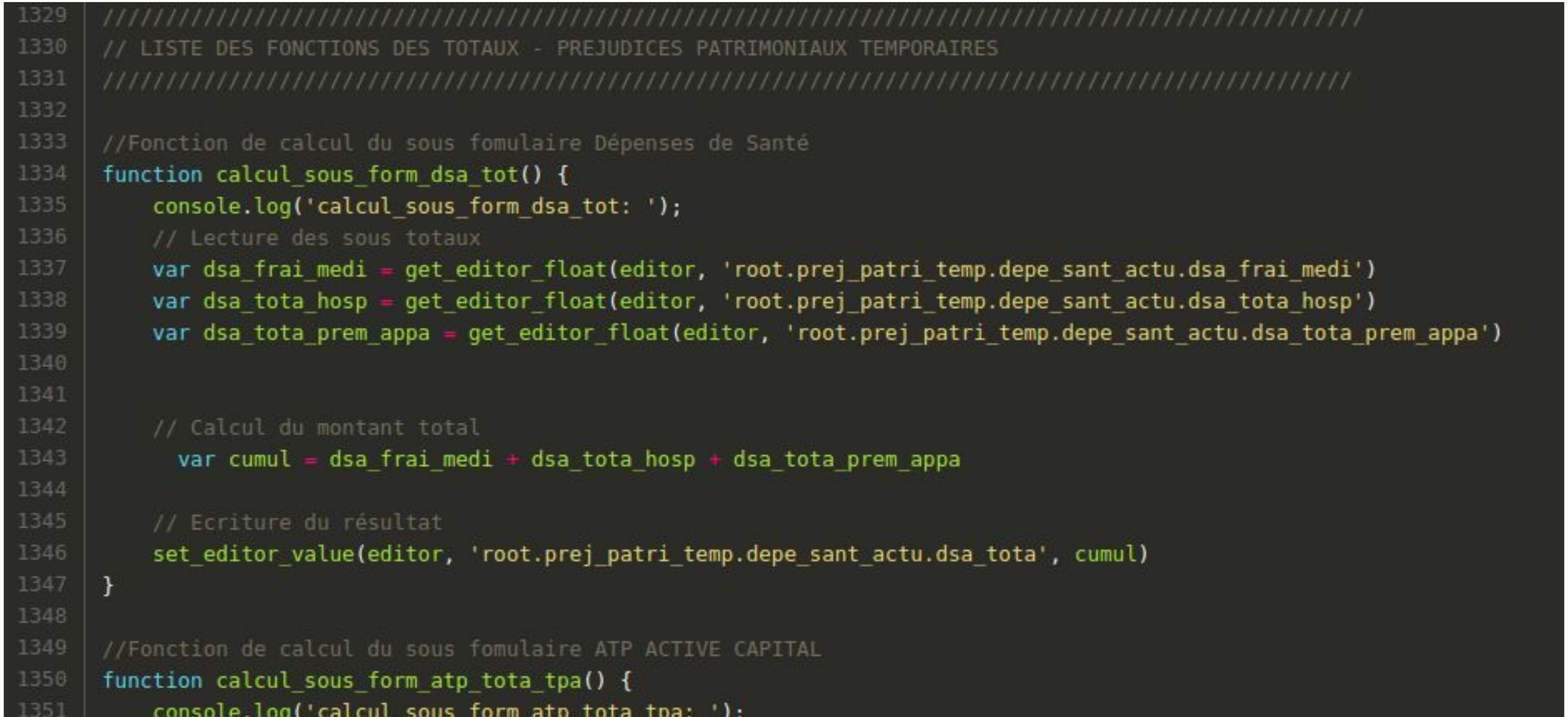
Hospitalisations	Nombre de jours	Coût journalier	Montant (Calculé)	Montant (Saisi)	Date de règlement
Autre	1	[redacted]	[redacted]	[redacted]	jj/mm/aaaa
IME	70	[redacted]	[redacted]	[redacted]	jj/mm/aaaa
IME	2730	[redacted]	[redacted]	[redacted]	jj/mm/aaaa

TOTAL Hospitalisations

[redacted]

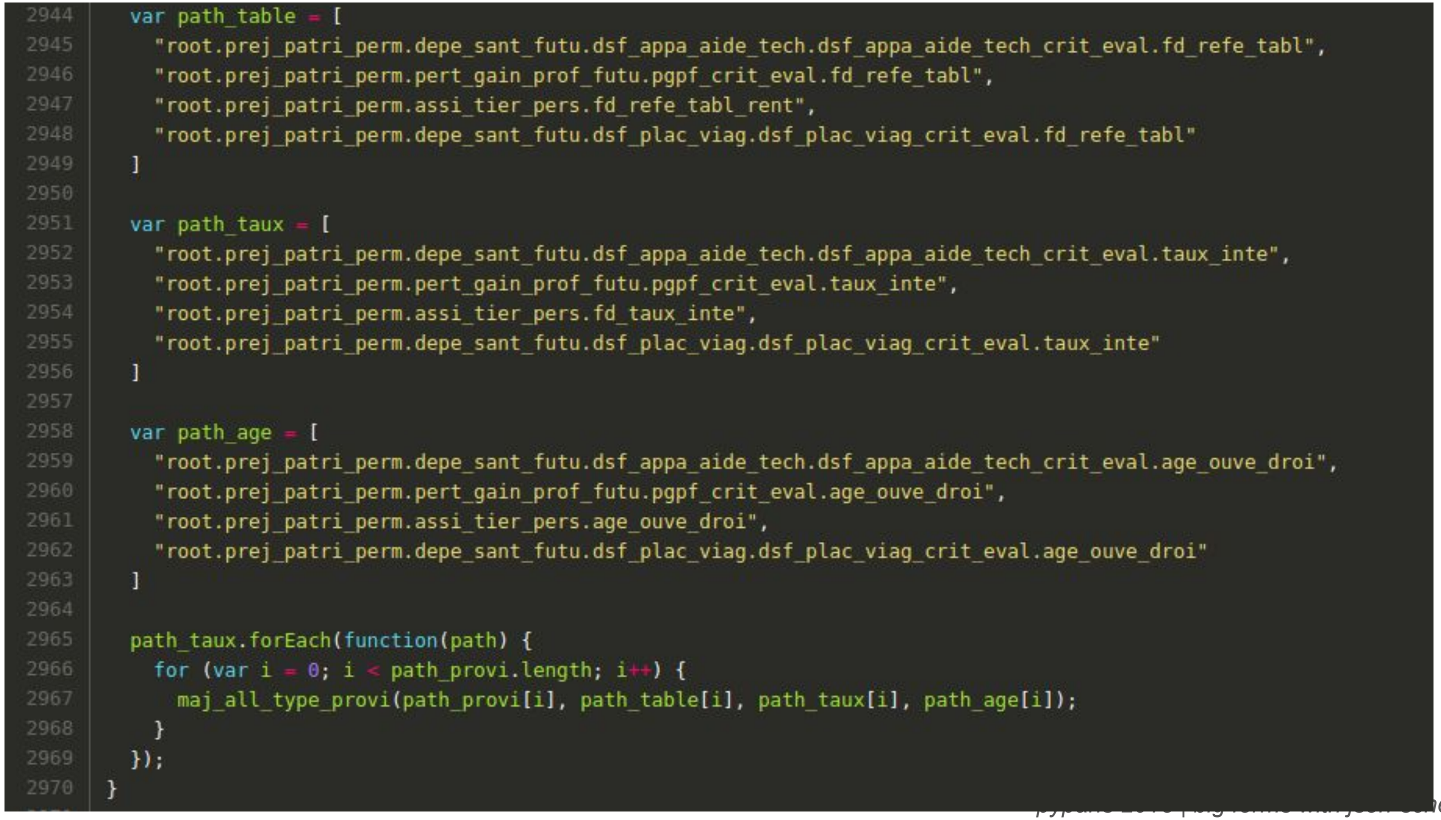
III. A. 2. 1er appareillage achat initial

It worked but at a cost ...



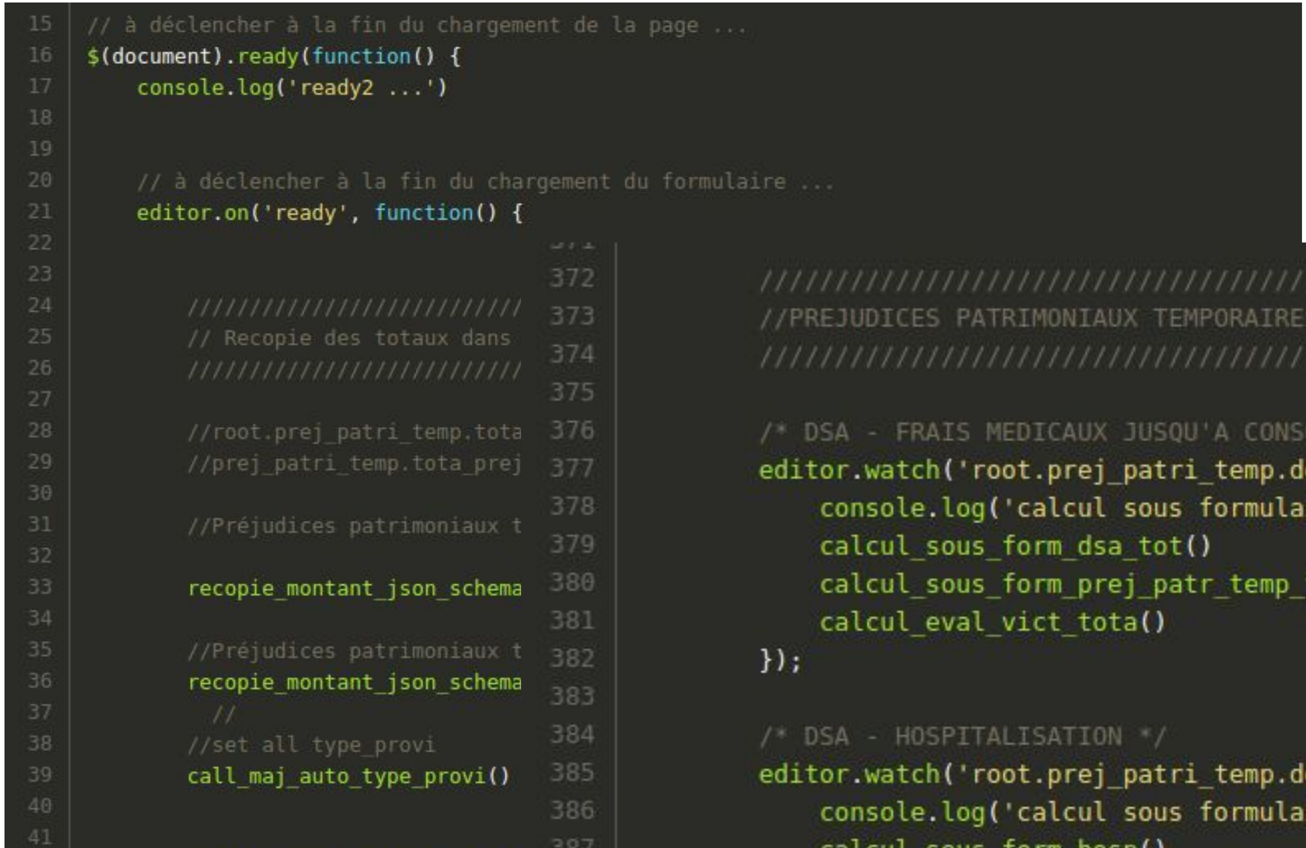
```
//////////////////////////////////////////////////////////////////////////////////////////
// LISTE DES FONCTIONS DES TOTAUX - PREJUDICES PATRIMONIAUX TEMPORAIRES
/////////////////////////////////////////////////////////////////////////////////////////

//Fonction de calcul du sous fomulaire Dépenses de Santé
function calcul_sous_form_dsa_tot() {
    console.log('calcul_sous_form_dsa_tot: ');
    // Lecture des sous totaux
    var dsa_frai_medi = get_editor_float(editor, 'root.prej_patri_temp.depe_sant_actu.dsa_frai_medi')
    var dsa_tota_hosp = get_editor_float(editor, 'root.prej_patri_temp.depe_sant_actu.dsa_tota_hosp')
    var dsa_tota_prem_appa = get_editor_float(editor, 'root.prej_patri_temp.depe_sant_actu.dsa_tota_prem_appa')


    // Calcul du montant total
      var cumul = dsa_frai_medi + dsa_tota_hosp + dsa_tota_prem_appa

    // Ecriture du résultat
    set_editor_value(editor, 'root.prej_patri_temp.depe_sant_actu.dsa_tota', cumul)
}

//Fonction de calcul du sous fomulaire ATP ACTIVE CAPITAL
function calcul_sous_form_atp_tota_tpa() {
    console.log('calcul_sous_form_atp_tota_tpa: ');
```

3000 LOC of clumpsy javascript business rules

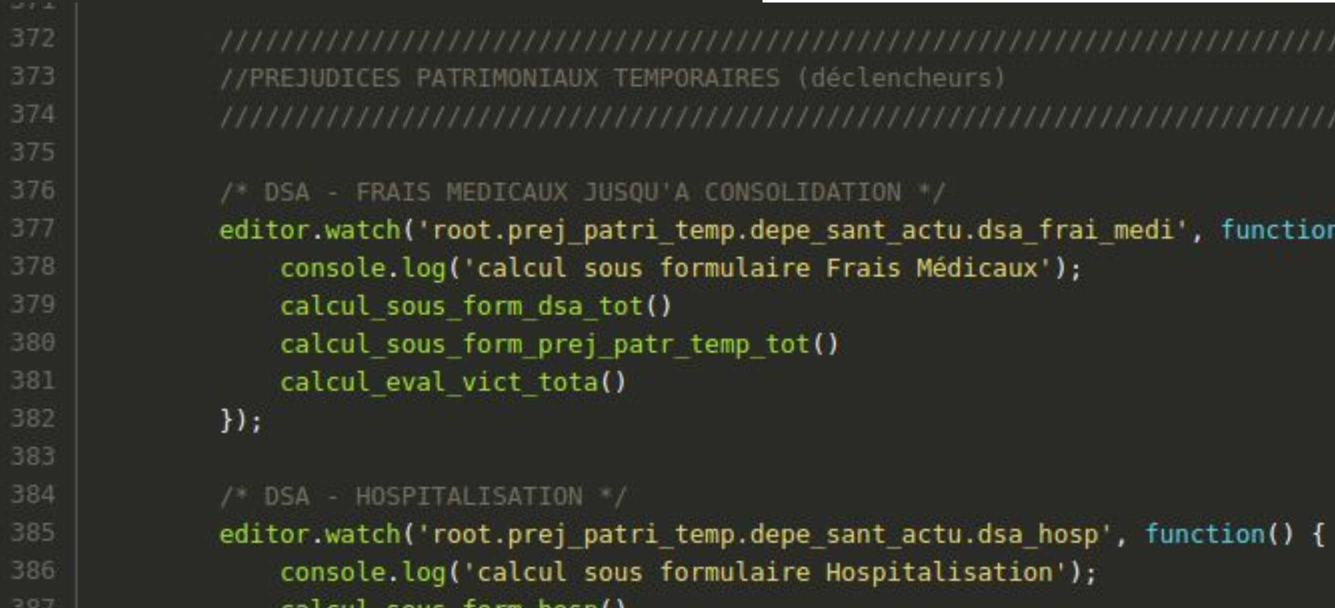


```
  var path_table = [
    "root.prej_patri_perm.depe_sant_futu.dsf_appa_aide_tech.dsf_appa_aide_tech_crit_eval.fd_refe_tabl",
    "root.prej_patri_perm.pert_gain_prof_futu.pgpf_crit_eval.fd_refe_tabl",
    "root.prej_patri_perm.assi_tier_pers.fd_refe_tabl_rent",
    "root.prej_patri_perm.depe_sant_futu.dsf_plac_viag.dsf_plac_viag_crit_eval.fd_refe_tabl"
  ]

  var path_taux = [
    "root.prej_patri_perm.depe_sant_futu.dsf_appa_aide_tech.dsf_appa_aide_tech_crit_eval.taux_inte",
    "root.prej_patri_perm.pert_gain_prof_futu.pgpf_crit_eval.taux_inte",
    "root.prej_patri_perm.assi_tier_pers.fd_taux_inte",
    "root.prej_patri_perm.depe_sant_futu.dsf_plac_viag.dsf_plac_viag_crit_eval.taux_inte"
  ]

  var path_age = [
    "root.prej_patri_perm.depe_sant_futu.dsf_appa_aide_tech.dsf_appa_aide_tech_crit_eval.age_ouve_droi",
    "root.prej_patri_perm.pert_gain_prof_futu.pgpf_crit_eval.age_ouve_droi",
    "root.prej_patri_perm.assi_tier_pers.age_ouve_droi",
    "root.prej_patri_perm.depe_sant_futu.dsf_plac_viag.dsf_plac_viag_crit_eval.age_ouve_droi"
  ]

  path_taux.forEach(function(path) {
    for (var i = 0; i < path_provi.length; i++) {
      maj_all_type_provi(path_provi[i], path_table[i], path_taux[i], path_age[i]);
    }
  });
}
```

Don’t ask non-dev to mess up with javascript



```
// à déclencher à la fin du chargement de la page ...
$(document).ready(function() {
    console.log('ready2 ...')


    // à déclencher à la fin du chargement du formulaire ...
    editor.on('ready', function() {


        ///////////////////////////
        // Recopie des totaux dans
        ///////////////////////////

        //root.prej_patri_temp.tota
        //prej_patri_temp.tota_prej

        //Préjudices patrimoniaux t

        recopie_montant_json_schema

        //Préjudices patrimoniaux t
        recopie_montant_json_schema
          //
        //set all type_provi
        call_maj_auto_type_provi()
```



```
        ////////////////////////////////////////////////////////////////////////
        //PREJUDICES PATRIMONIAUX TEMPORAIRES (déclencheurs)
        ////////////////////////////////////////////////////////////////////////

        /* DSA - FRAIS MEDICAUX JUSQU'A CONSOLIDATION */
        editor.watch('root.prej_patri_temp.depe_sant_actu.dsa_frai_medi', functio
            console.log('calcul sous formulaire Frais Médicaux');
            calcul_sous_form_dsa_tot()
            calcul_sous_form_prej_patr_temp_tot()
            calcul_eval_vict_tota()
        });

        /* DSA - HOSPITALISATION */
        editor.watch('root.prej_patri_temp.depe_sant_actu.dsa_hosp', function() {
            console.log('calcul sous formulaire Hospitalisation');
```

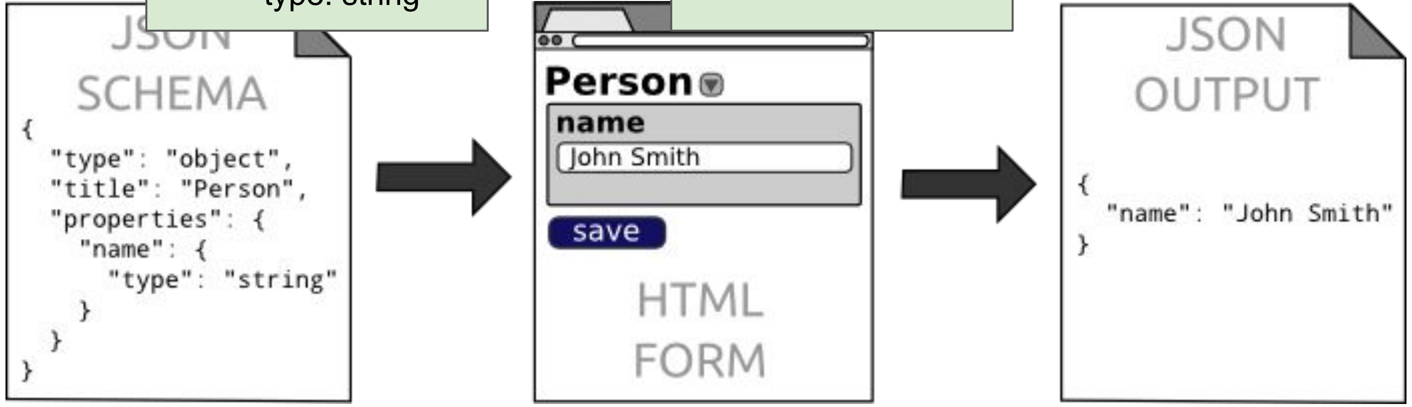
Hiding the javascript quirks with python

Move from JSON
to YAML format :

```
- type: object
- title: Person
- properties:
    - name:
        - type: string
```

Rewrite the business
rules from javacript
to python.

Run them on the
browser with
Transcrypt.

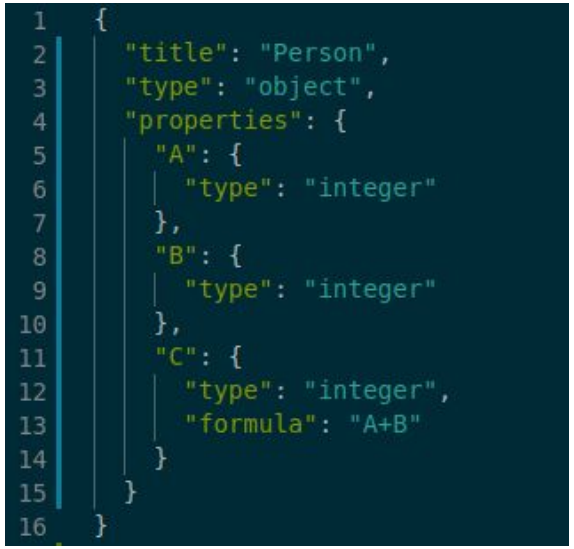


```
{
  "type": "object",
  "title": "Person",
  "properties": {
    "name": {
      "type": "string"
    }
  }
}
```

```
{
  "name": "John Smith"
}
```

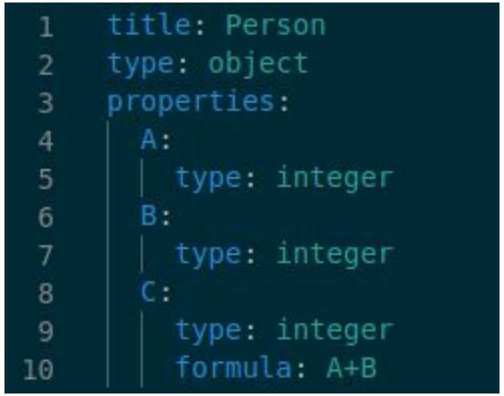
Better, stronger, faster, shorter

json-schema in JSON



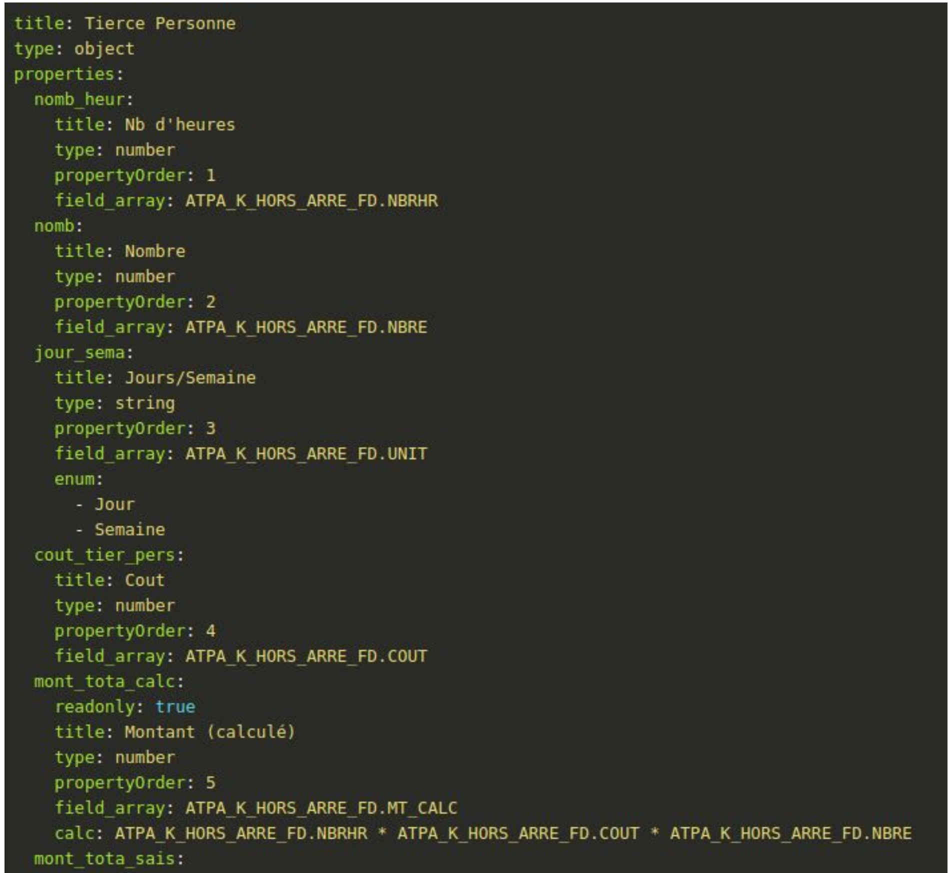
```
{
  "title": "Person",
  "type": "object",
  "properties": {
    "A": {
      "type": "integer"
    },
    "B": {
      "type": "integer"
    },
    "C": {
      "type": "integer",
      "formula": "A+B"
    }
  }
}
```

json-schema in YAML

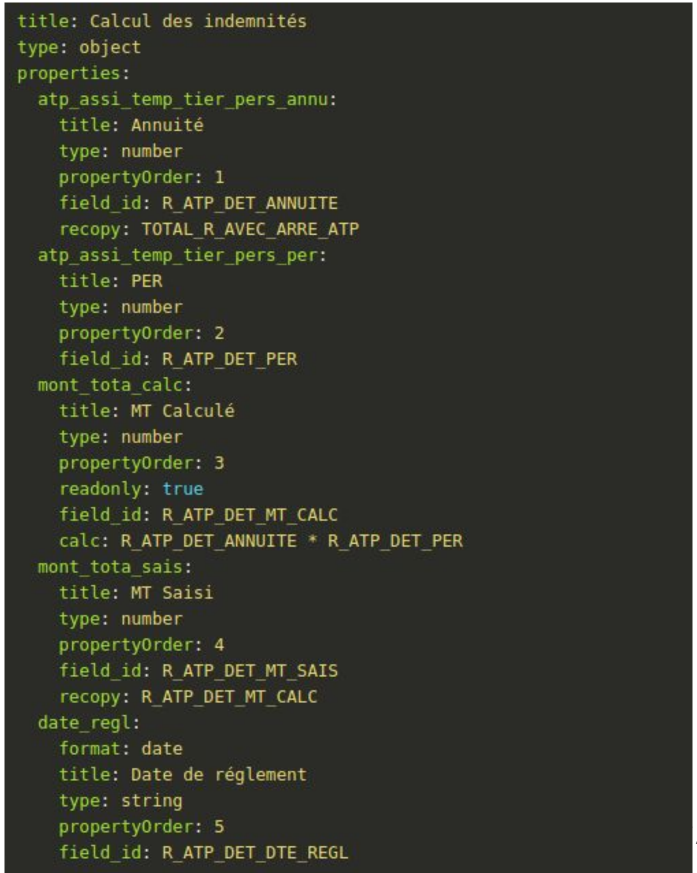


```
title: Person
type: object
properties:
  A:
    type: integer
  B:
    type: integer
  C:
    type: integer
    formula: A+B
```

Better, stronger, faster, shorter



```
title: Tierce Personne
type: object
properties:
  nomb_heur:
    title: Nb d'heures
    type: number
    propertyOrder: 1
    field_array: ATPA_K_HORS_ARRE_FD.NBRHR
  nomb:
    title: Nombre
    type: number
    propertyOrder: 2
    field_array: ATPA_K_HORS_ARRE_FD.NBRE
  jour_sema:
    title: Jours/Semaine
    type: string
    propertyOrder: 3
    field_array: ATPA_K_HORS_ARRE_FD.UNIT
    enum:
      - Jour
      - Semaine
  cout_tier_pers:
    title: Cout
    type: number
    propertyOrder: 4
    field_array: ATPA_K_HORS_ARRE_FD.COUT
  mont_tota_calc:
    readonly: true
    title: Montant (calculé)
    type: number
    propertyOrder: 5
    field_array: ATPA_K_HORS_ARRE_FD.MT_CALC
    calc: ATPA_K_HORS_ARRE_FD.NBRHR * ATPA_K_HORS_ARRE_FD.COUT * ATPA_K_HORS_ARRE_FD.NBRE
  mont_tota_sais:
```



```
title: Calcul des indemnités
type: object
properties:
  atp_assi_temp_tier_pers_annu:
    title: Annuité
    type: number
    propertyOrder: 1
    field_id: R_ATP_DET_ANNUITE
    recopy: TOTAL_R_AVEC_ARRE_ATP
  atp_assi_temp_tier_pers_per:
    title: PER
    type: number
    propertyOrder: 2
    field_id: R_ATP_DET_PER
  mont_tota_calc:
    title: MT Calculé
    type: number
    propertyOrder: 3
    readonly: true
    field_id: R_ATP_DET_MT_CALC
    calc: R_ATP_DET_ANNUITE * R_ATP_DET_PER
  mont_tota_sais:
    title: MT Saisi
    type: number
    propertyOrder: 4
    field_id: R_ATP_DET_MT_SAIS
    recopy: R_ATP_DET_MT_CALC
  date_regl:
    format: date
    title: Date de réglement
    type: string
    propertyOrder: 5
    field_id: R_ATP_DET_DTE_REGL
```

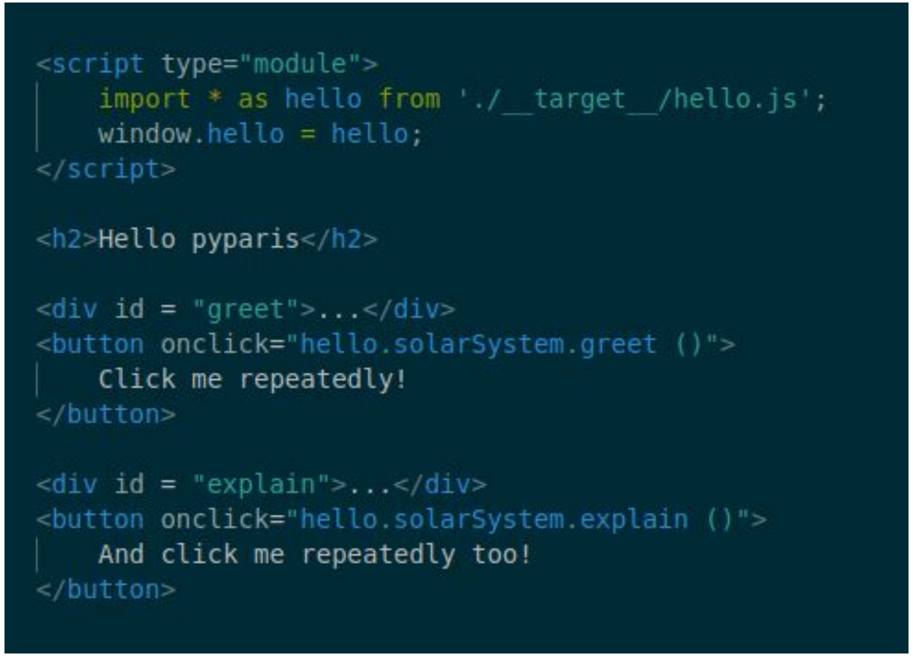
Feedback of our Transcrypt experience

1. Easy Transcrypt setup
2. Accessing DOM and JS objects
3. Calling JS from python and python from JS
4. eval() missing
5. Python object overloading
6. Example formulas
7. Unit tests with pytest
8. End-to-end tests with pytest/splinter/selenium
9. Debugging with or without sourcemap
10. Watch files for transpilation
11. Transcrypt overhead
12. Transcrypt alternatives

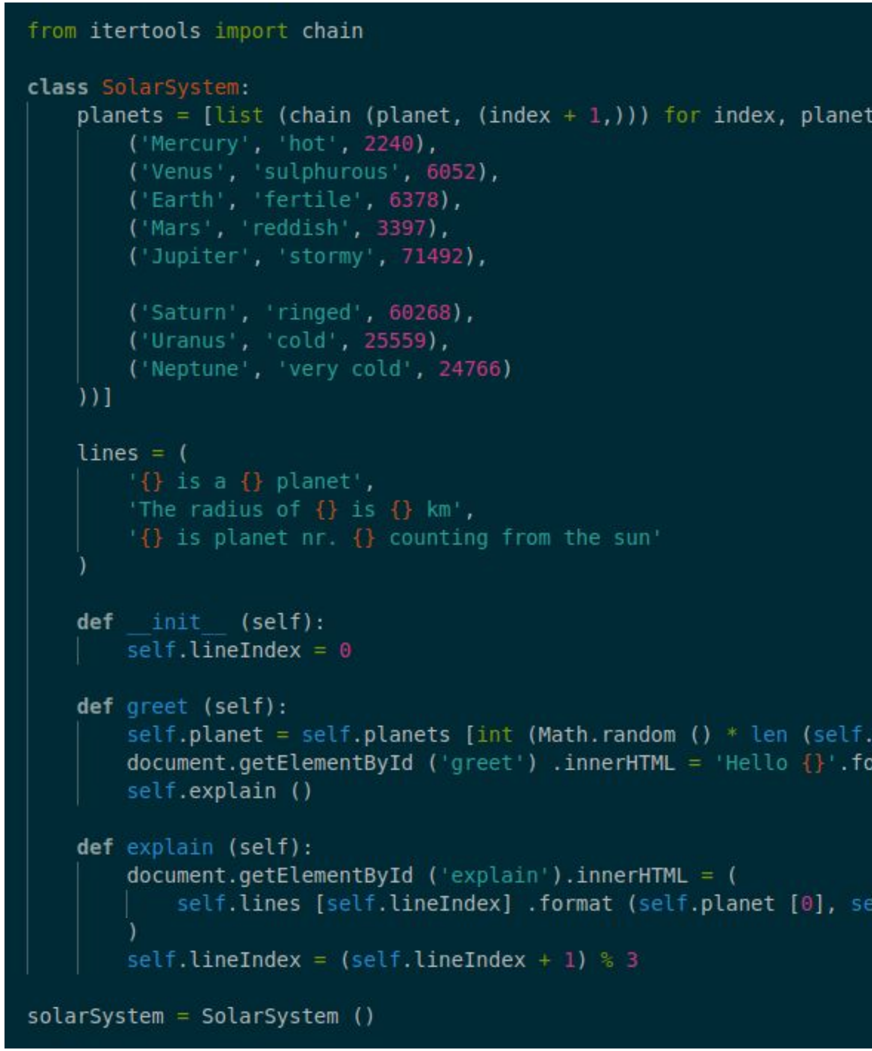
Easy Transcrypt setup

`$ pip install transcrypt`	**<< install** (+java for clojure minification)
`$ transcrypt hello`	**<< transpile hello.py to javascript**
`$ python3 -m http.server`	**<< serve static content**

Accessing DOM and JS objects

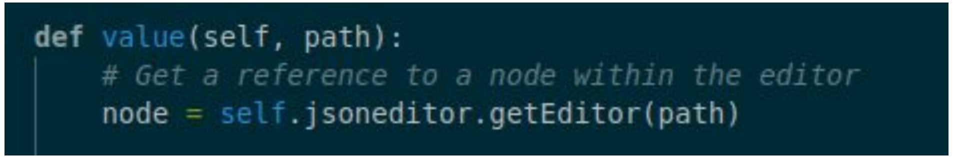


```html
<script type="module">
    import * as hello from './__target__/hello.js';
    window.hello = hello;
</script>

<h2>Hello pyparis</h2>

<div id = "greet">...</div>
<button onclick="hello.solarSystem.greet ()">
    Click me repeatedly!
</button>

<div id = "explain">...</div>
<button onclick="hello.solarSystem.explain ()">
    And click me repeatedly too!
</button>
```

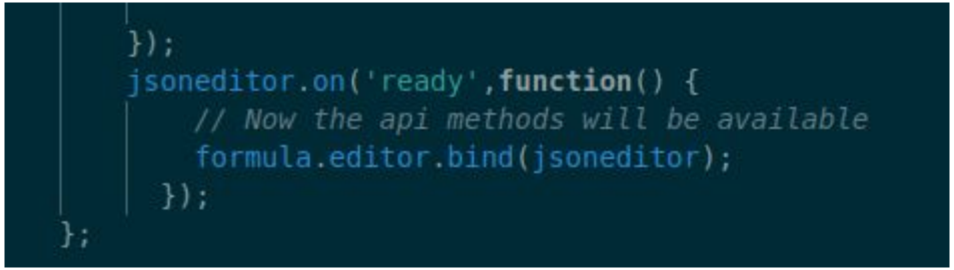


```python
from itertools import chain

class SolarSystem:
    planets = [list (chain (planet, (index + 1,))) for index, planet
        ('Mercury', 'hot', 2240),
        ('Venus', 'sulphurous', 6052),
        ('Earth', 'fertile', 6378),
        ('Mars', 'reddish', 3397),
        ('Jupiter', 'stormy', 71492),

        ('Saturn', 'ringed', 60268),
        ('Uranus', 'cold', 25559),
        ('Neptune', 'very cold', 24766)
    ))]

    lines = (
        '{} is a {} planet',
        'The radius of {} is {} km',
        '{} is planet nr. {} counting from the sun'
    )

    def __init__ (self):
        self.lineIndex = 0

    def greet (self):
        self.planet = self.planets [int (Math.random () * len (self.
        document.getElementById ('greet') .innerHTML = 'Hello {}'.fo
        self.explain ()

    def explain (self):
        document.getElementById ('explain').innerHTML = (
            self.lines [self.lineIndex] .format (self.planet [0], se
        )
        self.lineIndex = (self.lineIndex + 1) % 3

solarSystem = SolarSystem ()
```

Calling JS from python and python from JS



```python
def value(self, path):
    # Get a reference to a node within the editor
    node = self.jsoneditor.getEditor(path)
```



```javascript
        });
        jsoneditor.on('ready',function() {
            // Now the api methods will be available
            formula.editor.bind(jsoneditor);
          });
  };
```

eval() missing

Evaluating our formulas is easy with eval()

The single disappointment in our experiment :

eval() is not implemented in Transcrypt

You must use the transpiler server-side only.

So we had to parse and evaluate our formulas in python.

Python object overloading

Transcrypt is very close to Python regarding subclassing, overloading, compositing objects.

We implemented a simple formula parser and a schema/document walker.

All the python tricks we needed worked :

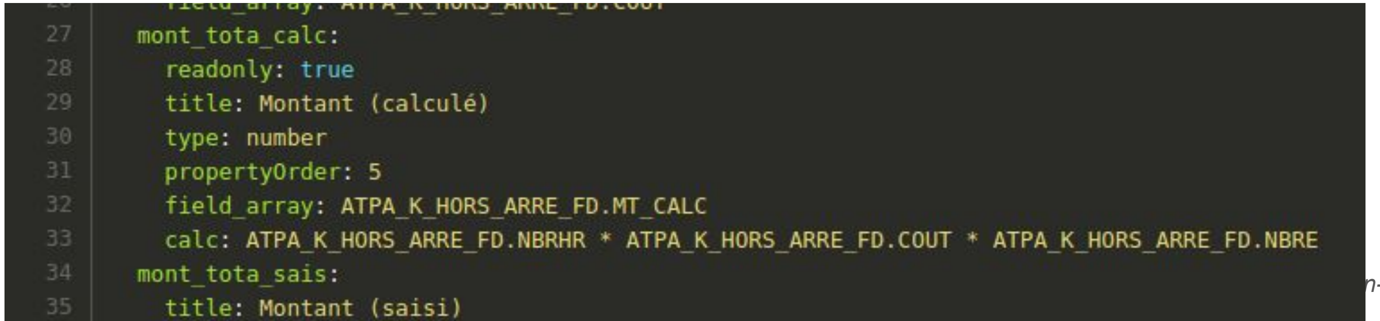
`__get__`, `__missing__`, `__setitem__`, `__iter__`, `__contains__`, ...

Example formulas

Simple formula : AMOUNT * QTY
refering to nearby fields

Dot notation formula : sum(HOSP.NB * HOSP.AMOUNT)
refering to array and doing matrix operation

Custom function : my_special_pricer(x, y, z)
defined in Python (Transcrypt)

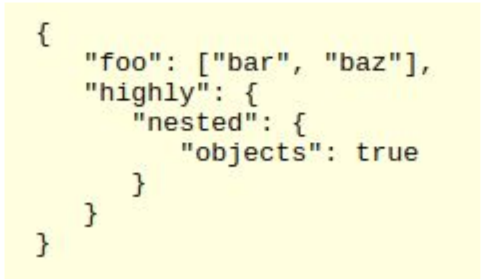


```
  field_array: ATPA_K_HORS_ARRE_FD.COUT
mont_tota_calc:
  readonly: true
  title: Montant (calculé)
  type: number
  propertyOrder: 5
  field_array: ATPA_K_HORS_ARRE_FD.MT_CALC
  calc: ATPA_K_HORS_ARRE_FD.NBRHR * ATPA_K_HORS_ARRE_FD.COUT * ATPA_K_HORS_ARRE_FD.NBRE
mont_tota_sais:
  title: Montant (saisi)
```

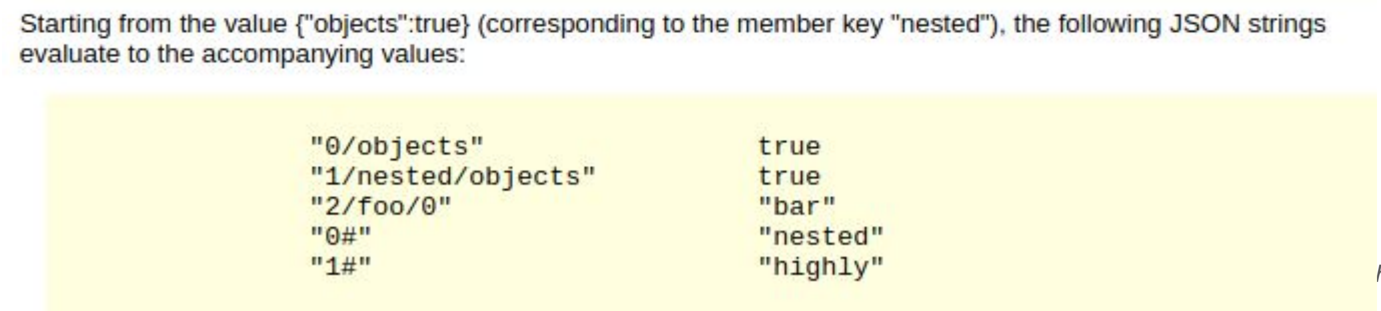
Relative json pointer support

We plan to support the draft proposal relative-json-pointer that will help for some complex cases of relative references.

Example :



```
{
   "foo": ["bar", "baz"],
   "highly": {
      "nested": {
         "objects": true
      }
   }
}
```



Starting from the value {"objects":true} (corresponding to the member key "nested"), the following JSON strings evaluate to the accompanying values:

```
"0/objects"             true
"1/nested/objects"      true
"2/foo/0"               "bar"
"0#"                    "nested"
"1#"                    "highly"
```

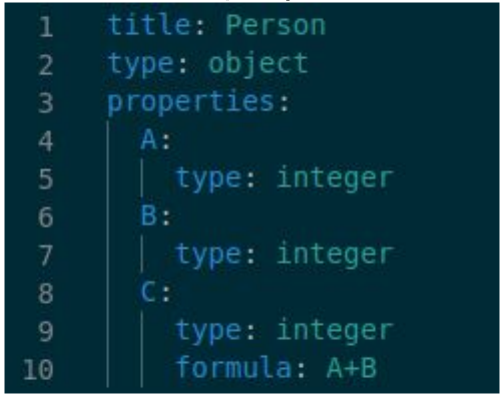
Unit tests with pytest

Formulas in schema are tested.

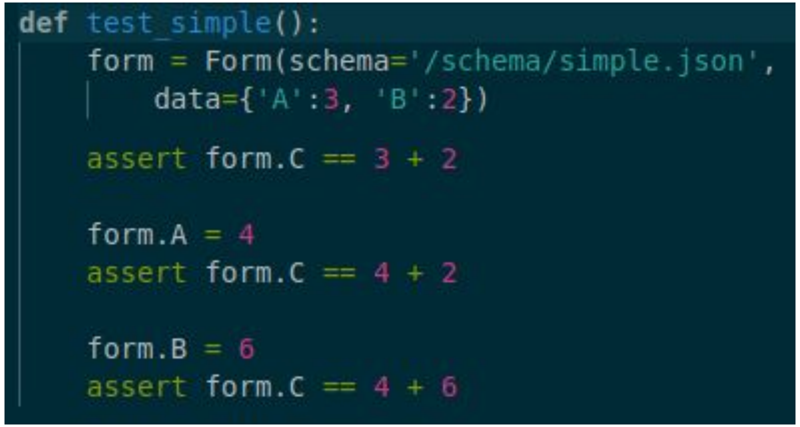
Dozens of tests, easely readable and writable by the business experts.

Run on CPython.

/schema/simple.yaml

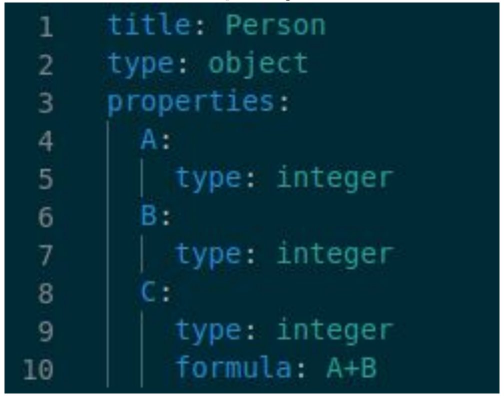


```yaml
title: Person
type: object
properties:
  A:
    type: integer
  B:
    type: integer
  C:
    type: integer
    formula: A+B
```



```python
def test_simple():
    form = Form(schema='/schema/simple.json',
        data={'A':3, 'B':2})

    assert form.C == 3 + 2

    form.A = 4
    assert form.C == 4 + 2

    form.B = 6
    assert form.C == 4 + 6
```

/schema/simple.yaml



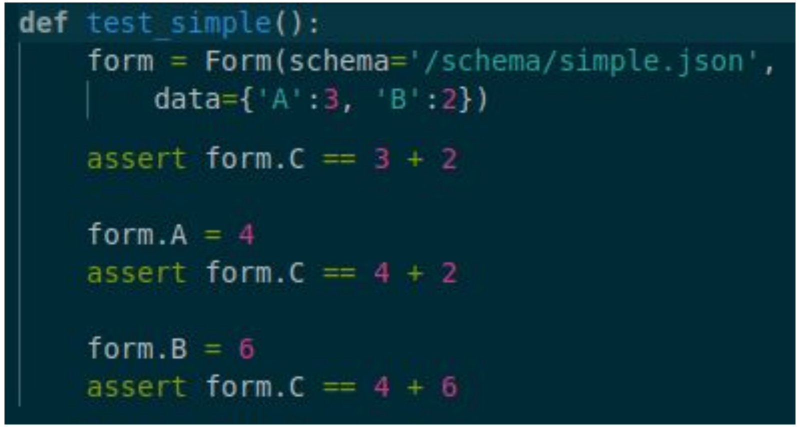
```yaml
title: Person
type: object
properties:
  A:
    type: integer
  B:
    type: integer
  C:
    type: integer
    formula: A+B
```

End-to-end tests with selenium

Same tests !

Automatically transpiled to Javascript
and run on a real broswer with selenium, splinter
and pytest.

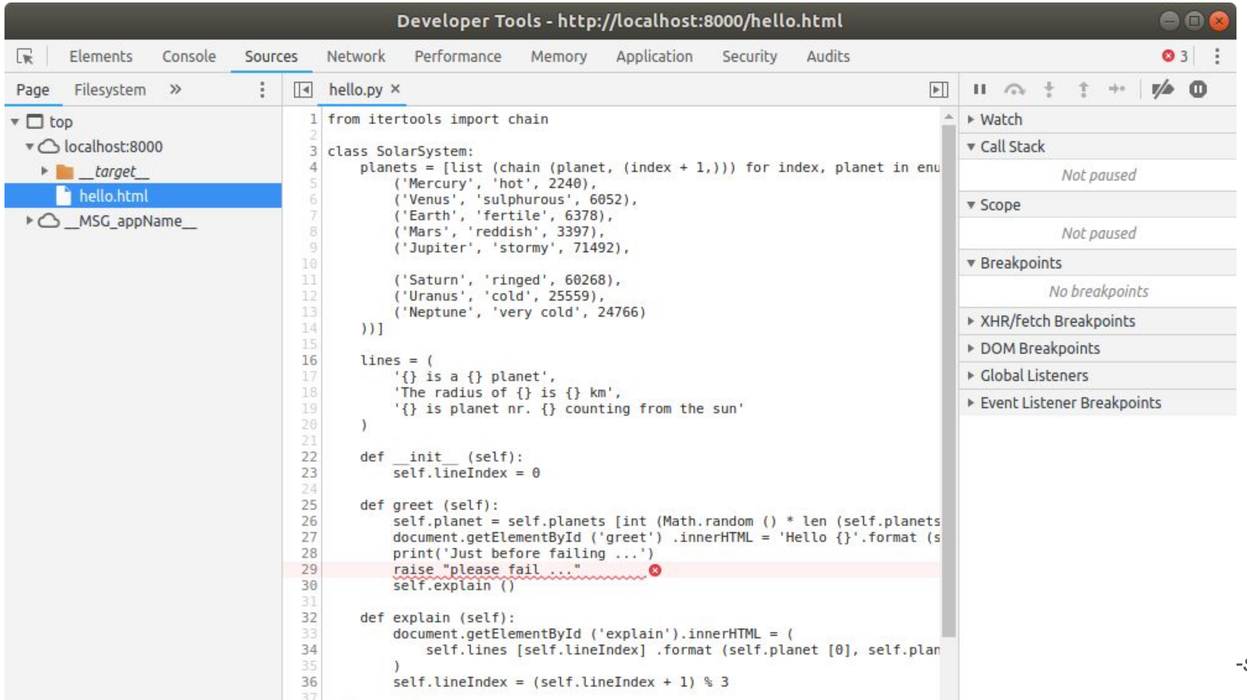
Chrome headless mode for running on CI jobs.



```python
def test_simple():
    form = Form(schema='/schema/simple.json',
                data={'A':3, 'B':2})

    assert form.C == 3 + 2

    form.A = 4
    assert form.C == 4 + 2

    form.B = 6
    assert form.C == 4 + 6
```

Sourcemap debugging

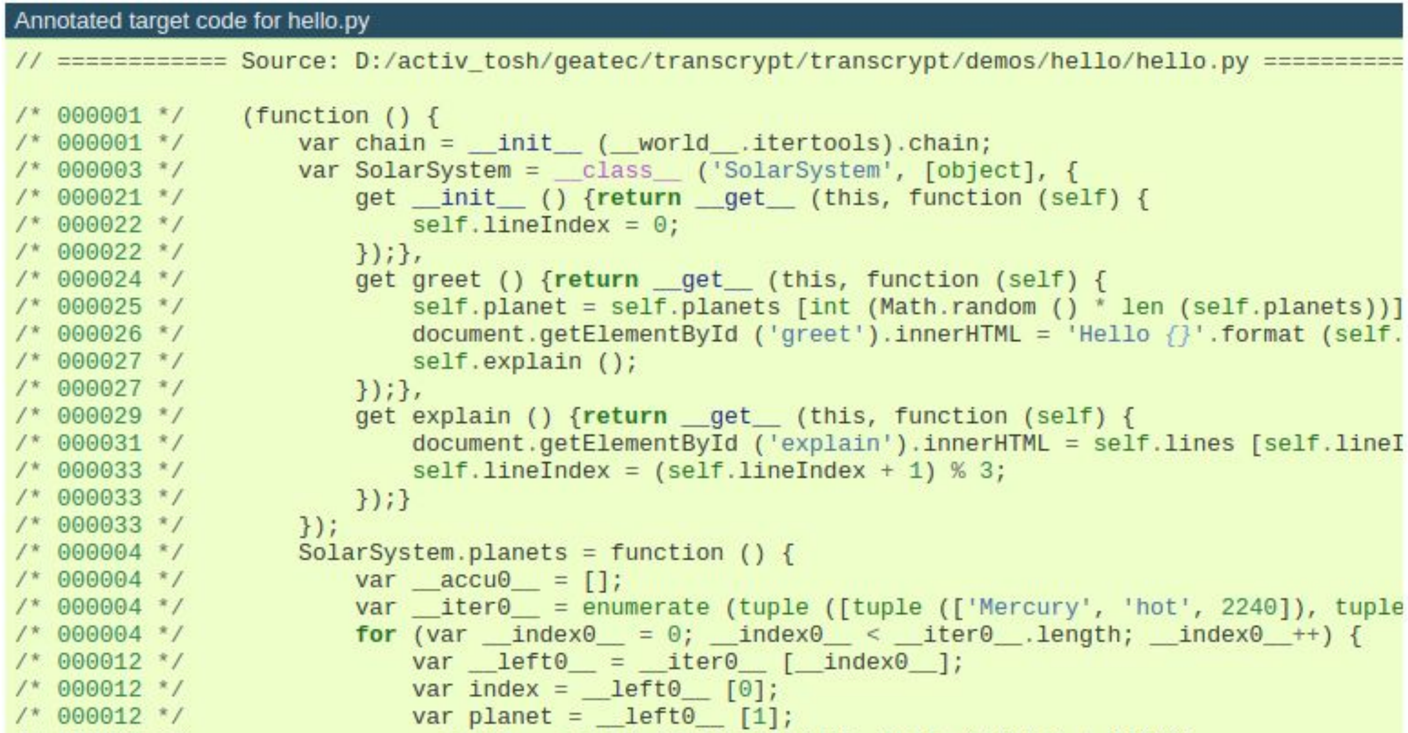
transcrypt -m hello



```
from itertools import chain

class SolarSystem:
    planets = [list (chain (planet, (index + 1,))) for index, planet in enu
        ('Mercury', 'hot', 2240),
        ('Venus', 'sulphurous', 6052),
        ('Earth', 'fertile', 6378),
        ('Mars', 'reddish', 3397),
        ('Jupiter', 'stormy', 71492),

        ('Saturn', 'ringed', 60268),
        ('Uranus', 'cold', 25559),
        ('Neptune', 'very cold', 24766)
    ))]

    lines = (
        '{} is a {} planet',
        'The radius of {} is {} km',
        '{} is planet nr. {} counting from the sun'
    )

    def __init__ (self):
        self.lineIndex = 0

    def greet (self):
        self.planet = self.planets [int (Math.random () * len (self.planets
        document.getElementById ('greet') .innerHTML = 'Hello {}'.format (s
        print('Just before failing ...')
        raise "please fail ..."
        self.explain ()

    def explain (self):
        document.getElementById ('explain').innerHTML = (
            self.lines [self.lineIndex] .format (self.planet [0], self.plan
        )
        self.lineIndex = (self.lineIndex + 1) % 3
```

Debugging without sourcemap

transcrypt -a -n hello



Annotated target code for hello.py

```
// ============ Source: D:/activ_tosh/geatec/transcrypt/transcrypt/demos/hello/hello.py ==========

/* 000001 */    (function () {
/* 000001 */        var chain = __init__ (__world__.itertools).chain;
/* 000003 */        var SolarSystem = __class__ ('SolarSystem', [object], {
/* 000021 */            get __init__ () {return __get__ (this, function (self) {
/* 000022 */                self.lineIndex = 0;
/* 000022 */            });},
/* 000024 */            get greet () {return __get__ (this, function (self) {
/* 000025 */                self.planet = self.planets [int (Math.random () * len (self.planets))]
/* 000026 */                document.getElementById ('greet').innerHTML = 'Hello {}'.format (self.
/* 000027 */                self.explain ();
/* 000027 */            });},
/* 000029 */            get explain () {return __get__ (this, function (self) {
/* 000031 */                document.getElementById ('explain').innerHTML = self.lines [self.lineI
/* 000033 */                self.lineIndex = (self.lineIndex + 1) % 3;
/* 000033 */            });}
/* 000033 */        });
/* 000004 */        SolarSystem.planets = function () {
/* 000004 */            var __accu0__ = [];
/* 000004 */            var __iter0__ = enumerate (tuple ([tuple (['Mercury', 'hot', 2240]), tuple
/* 000004 */            for (var __index0__ = 0; __index0__ < __iter0__.length; __index0__++) {
/* 000012 */                var __left0__ = __iter0__ [__index0__];
/* 000012 */                var index = __left0__ [0];
/* 000012 */                var planet = __left0__ [1];
```

Watch file for transpilation

Static transpilation not an option in our case since users can change python source (schema formula)

Watch and transpile : run transcrypt again on any file change

    entr or inotify tools

Transcrypt overhead

The minified JavaScript code for each of your own modules is roughly just as large as the Python source code. On top of that there's a one time overhead of 20kB for Transcrypt's core and built-ins. Should you use the JavaScript 5 to 6 translator, that adds an extra 10kB. For larger projects, the overhead becomes negligeable. A project with a Python source of say 600kB tends to result in a dowload of about equal size. Moreover Python sourcecode for a certain application tends to be smaller than handwritten JavaScript source code for the same problem, due to language constructs like list comprehensions, but also due to facilities like class based OO and multiple inheritance. As far as speed is concerned, in most cases it is roughly equal to the speed of hand-written JavaScript. [..]

from Transcrypt's FAQ

Transcrypt alternatives

- transcrypt : transpiler, partial python support, numpy port
- rapidscript : transpiler, support eval() !
- brython : full python interpreter
- pyodide : WASM based
- batavia : python VM, run python bytecode, not source !

- pyjs : full python interpreter ?
- pypyjs : full python interpreter, emscripten/ASM based
- jiphy : transpiler, too limited

Being able to push Python in the browser helps us to add features to our automatically generated big forms.

Transcrypt saves us from the two languages pitfall in a critical part of our project. The overhead induced is negligeable in our case.

We are closing gaps between the front-end and back-end developement by sharing the same languages and test framework.

Python everywhere, really ?

So we get python on the browser and we’re happy with it.

Lonely trick or real trend ?

We think it’s a bold move for a good reason :

All the technologies are moving faster …

… but our brain is not !!

You can’t master many programming languages.
Non-developper can only learn a single trivial programming language.
Developpers and non-dev. must have a common programming language.
The toolset shared among the team must be as light as possible.

The Python language and eco-system is the best fit today.

Split the stack

Visible to anyone :

jupyter > YAML > pytest > xlwings
> python > json-schema > plotly > pandas

Must stay hidden except for devops :

transcrypt, cpython, anaconda, mongo, docker, kubernetes, flask, swagger, angular, bootstrap, caddy, docker-compose, pyinstaller, git, dash, python-pptx, dramatiq, secretary, gitlab, javascript, ...

Thank you !

philippe@phec.net

Image credits :

- Car top view by qubodup
- Motor vehicle accident illustration by oksmith
- Injured illustration by oksmith
- Head with brain silhouette illustration by monstara, GDJ