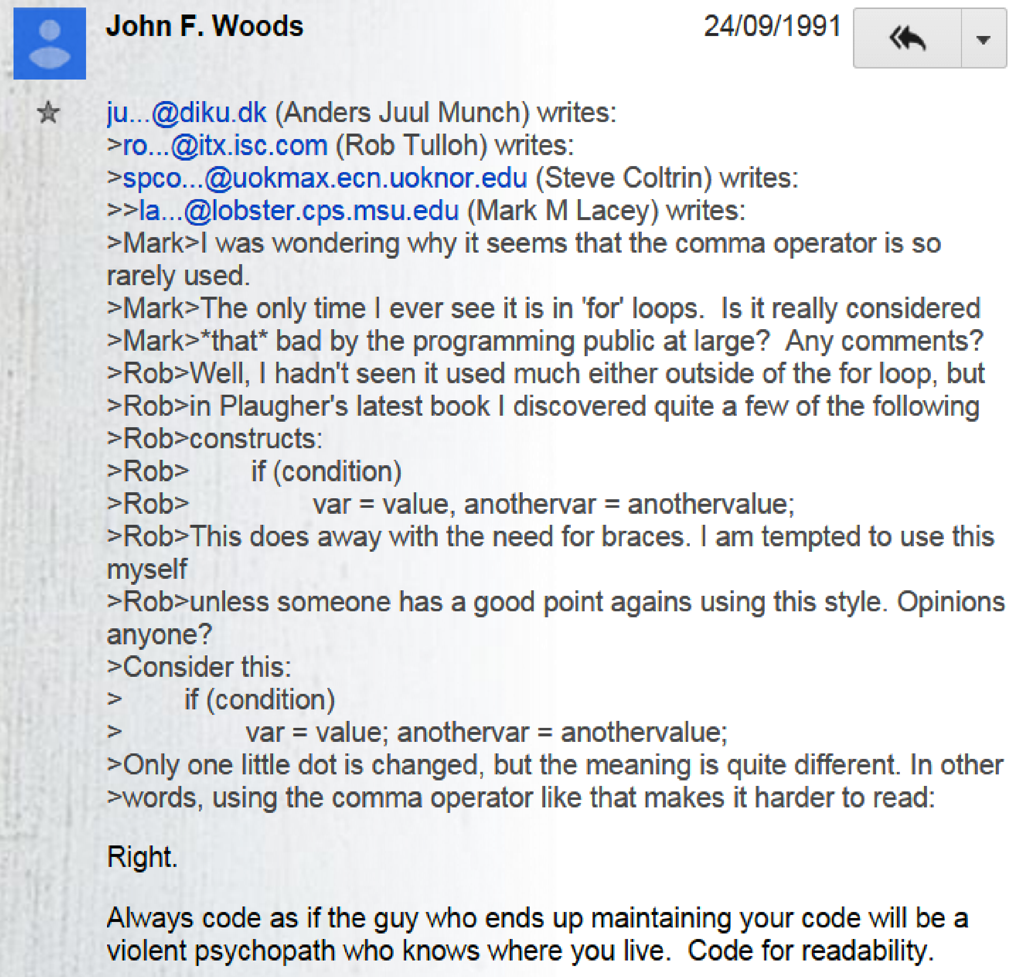
Advanced Web Technologies

Geoffray Bonnin

Exercise Sheet 2

Submitted their work: 36% (10 people)

- Not completed
  - Reinitiate the race: 60%
  - EPO: 50%
- Not well commented: 100%
- Alerts: 90%
- Used jQuery: 10% (1 student)



John F. Woods 24/09/1991

```
ju...@diku.dk (Anders Juul Munch) writes:
>ro...@itx.isc.com (Rob Tulloh) writes:
>spco...@uokmax.ecn.uoknor.edu (Steve Coltrin) writes:
>>la...@lobster.cps.msu.edu (Mark M Lacey) writes:
>Mark>I was wondering why it seems that the comma operator is so
rarely used.
>Mark>The only time I ever see it is in 'for' loops.  Is it really considered
>Mark>*that* bad by the programming public at large?  Any comments?
>Rob>Well, I hadn't seen it used much either outside of the for loop, but
>Rob>in Plaugher's latest book I discovered quite a few of the following
>Rob>constructs:
>Rob>       if (condition)
>Rob>               var = value, anothervar = anothervalue;
>Rob>This does away with the need for braces. I am tempted to use this
myself
>Rob>unless someone has a good point agains using this style. Opinions
anyone?
>Consider this:
>       if (condition)
>               var = value; anothervar = anothervalue;
>Only one little dot is changed, but the meaning is quite different. In other
>words, using the comma operator like that makes it harder to read:
```

Right.

Always code as if the guy who ends up maintaining your code will be a violent psychopath who knows where you live. Code for readability.

Good practices in JavaScript

(1) Avoid global variables

- Possible conflicts with other scripts
- Inside functions: **always** use `var` to declare them
- Global variables are not evil…

Good practices in JavaScript

(2) Avoid large functions

- Avoid loops with several levels, complex conditions, etc.
- Avoid too obvious comments
- Use subfunctions (except if these functions have too much parameters)
- Good indicator: functions of more than one page

Good practices in JavaScript

(3) Comments…

```
/**
 * Checks if a horizontal/vertical line passing through T intersects with
 * the line segment AB in a given direction from T.
 * @param {Object} T - The point T
 * @param {Object} A - The point A
 * @param {Object} B - The point B
 * @param {string} direction - The direction ("left", "right", "up", down")
 * @returns {boolean}
 */
function intersects(T, A, B, direction){
  if (direction == "left" || direction == "right"){
    // Horizontal line: yT must be between yA and yB
    if (A.y > B.y && T.y > A.y && B.y > T.y) return false;
    if (B.y > A.y && A.y > T.y && T.y > B.y) return false;

    // Is AB vertical?
    if (A.x == B.x){
      if (direction == "left"){
        return T.x <= A.x;
      }
      else{
        return T.x >= A.x;
      }
    }
    else{
      //Compute the x-coordinate of the intersection (the y-coordinate is yT)
      var x = (T.y - A.y) * (B.x - A.x) / (B.y - A.y) + A.x;
      if (direction == "left"){
        return x > xT;
      }
    …
```

JSDOC

Good practices in JavaScript

(4) Avoid alerts

- Alerts are cool: modal window in a few characters
- But obtrusive: they lock the browser
- Alert only when necessary
- If necessary, one can use modal dialogs from libraries
  - More homogeneous (from one browser to the other)
  - More beautiful

Correction Exercise Sheet 2

jQuery

jQuery

- JavaScript library
  - Very easy to learn
  - Designed to work on every browsers
- How to use jQuery
  - Can be downloaded: www.jquery.com
    (file *jquery-[version].js*)
  - Or directly integrated from the Web

```
<head>
  <script
src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.1/jquery.min.js">
  </script>
</head>
```

or

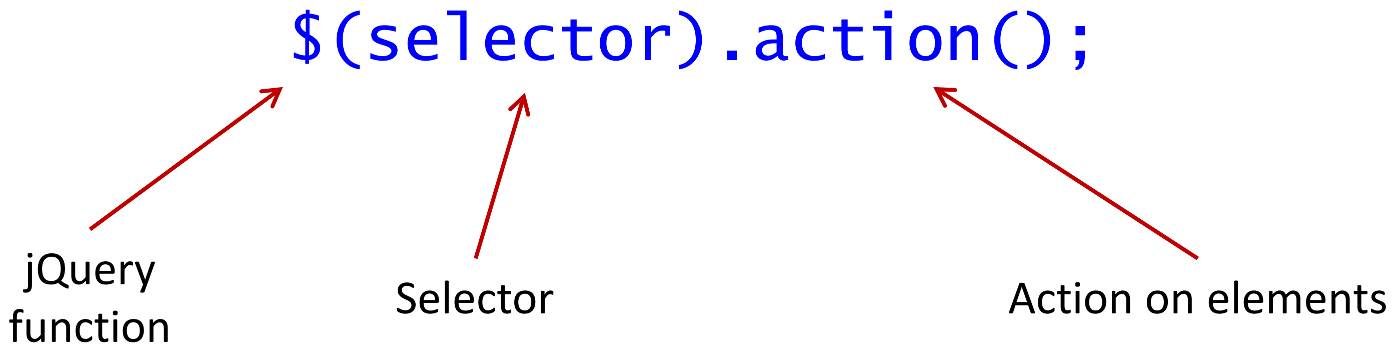
```
<head>
  <script src="http://ajax.aspnetcdn.com/ajax/jQuery/jquery-1.11.1.min.js">
  </script>
</head>
```

Syntax of jQuery

Three parts

- jQuery function
- Selector
- Action on selected elements

```
$(selector).action();
```



The jQuery function

Use the ‘$’ character

- JavaScripts accepts it as a valid character for variables
  ```
  var $ = 2;
  var $2 = 8;
  ```
- `$(…)` short
- Also easy to identify in the code
- Completely different from the PHP ‘$’

Selector

The ‘$’ character is always followed by parenthesis (function)

`$(`*`selector`*`)`

- Allows to select HTML elements in the page
- Use the CSS syntax

`$("a.externe")` $\longrightarrow$ every `<a class="externe">` elements

`$("#lala")` $\longrightarrow$ the element with the id “lala”

`$("a.externe, #lala")` $\longrightarrow$ every `<a class="externe">` **and** the element with id “lala”

Actions

Actions on selected elements

- Functions defined in the jQuery library

```
$("a.externe").css("text-decoration", "none");
```

→ deletes the decoration of all the `<a class="externe">`

- Returns the elements on which the action was applied:

```
$("a.externe").css("text-decoration", "none")
              .css("background-color", "blue");
```

Some selectors

Selector	Description
`$("*")`	Every elements
`$("p.haha")`	All paragraphs having `class="haha"`
`$("p:first")`	The first paragraph
`$("p:last")`	The last paragraph
`$("ul li:first")`	The first element of a list
`$("[href]")`	All elements having the attribute `href`

More selectors:
http://www.w3schools.com/jquery/jquery_ref_selectors.asp

Some functions (events)

- Mouse: `click()`, `dblclick()`, `mouseenter()`, `mouseleave()`, `mouseup()`, `mousedown()`, `hover()`
- Keyboard: `keypress()`, `keydown()`, `keyup()`
- Forms: `submit()`, `change()`, `focus()`, `blur()`
- Window: `load()`, `resize()`, `scroll()`, `unload()`

More event functions:
http://www.w3schools.com/jquery/jquery_ref_events.asp

Some other functions (effects)

Function	Description
`animate()`	Start an animation on the selected elements
`fadeIn()`	Make an element appear with a fade in effect
`hide()`	Hide the selected elements
`show()`	Show the selected elements
`slideDown()`	Make the selected elements appear with a slide down effect
`stop()`	Stop the effects on the selected elements

More effect functions:
http://www.w3schools.com/jquery/jquery_ref_effects.asp

Some other functions (content)

- Getters

`$("#id").html();`	*Content of the element with the id “id”*
`$("#id").text();`	*Text if the element with the id “id”*
`$("#edit").val();`	*Value in an input*

- Setters

```
$("#id").html("tralala <b>lala</b>");
$("#id").text("tralala lala");
$("#edit").val("42");
```

Attributes and CSS

- Getters

```
$("#id").attr("width");
$("#id").css("background-color");
```

- Setters

```
$("#id").attr("width", "150");
$("#id").css("background-color", "red");
$("#id").css({"background-color":"red", "width":"400px"});
```

DOM tree processing

Action	Description
`$("#son").parent()`	The parent of the *"son"* element (father)
`$("#son").parents()`	Set of ancestors (father, grandfather, document)
`$("#son").parentsUntil("#grandfather");`	(father, grandfather)
`$("#grandfather").children();`	(father)
`$("#father").children("span#son");`	(son)
`$("#grandfather").find("son");`	(son)

```
<div id="grandfather">
  <p id="father">
    <span id="son">bar</span>
    <span id="siblings">foo</span>
  </p>
</div>
```

DOM tree processing

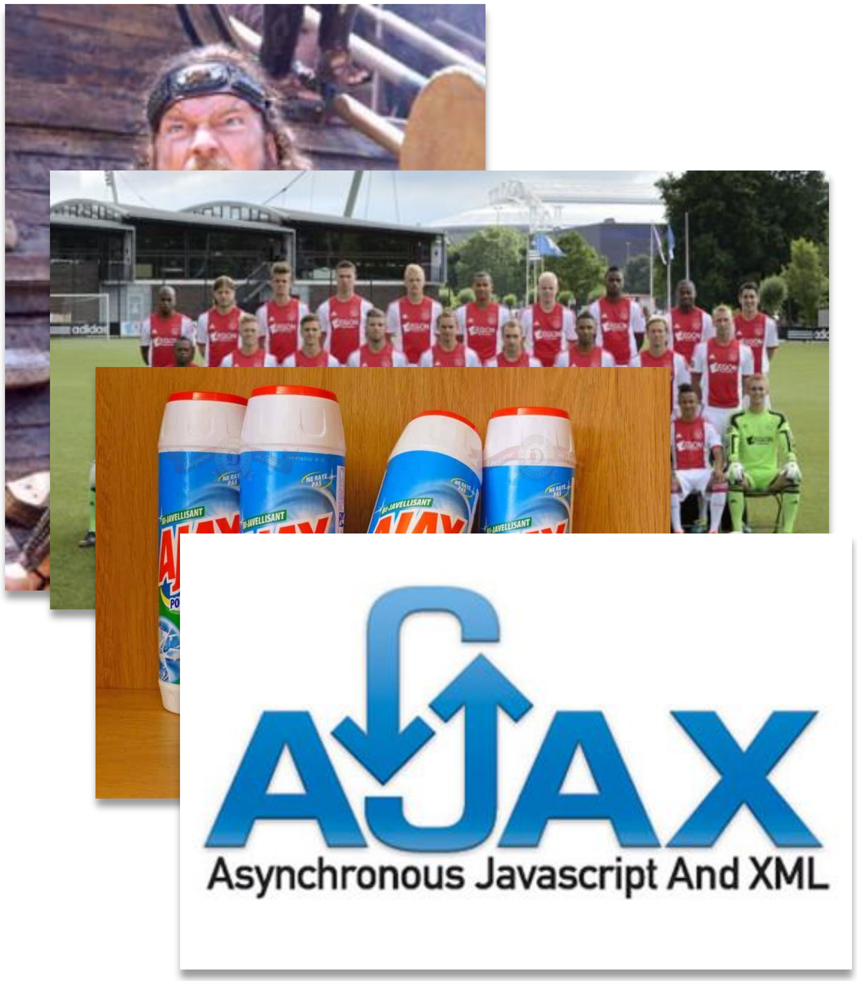
Action	Description
`$("#son").siblings()`	(siblings)
`$("#son").next()`	siblings
`$("span").not("#siblings");`	(son)
`$("#father").remove()`	Remove the element and all its descendants
`$("#father").empty()`	Remove all the descendants of te element

```
<div id="grandfather">
  <p id="father">
    <span id="son">bar</span>
    <span id="siblings">foo</span>
  </p>
</div>
```

AJAX, JSON and XML

Ajax history

- Mythology
  - Hero of the Trojan War
  - Number 2 after Achilles
- Football (1894)
  - Dutch football club
  - Several times the winner of the Champions League (70’s and 80’s)
- Cleaning product
- Web technology (early 2000’s)

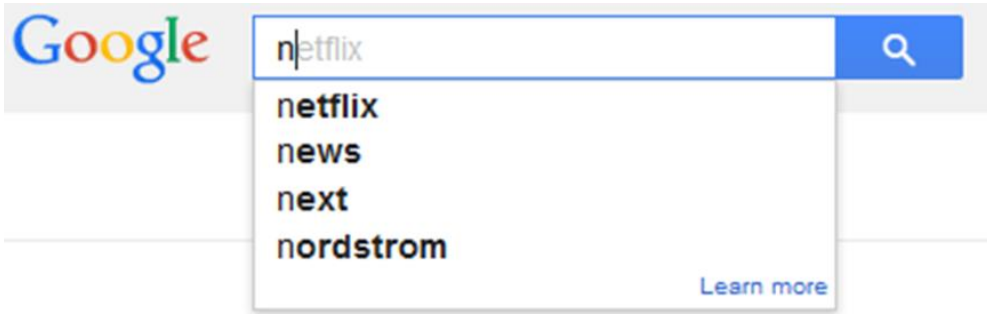


Ajax, web technology

- AJAX: *Asynchronous JavaScript and XML*
- Allow
  - Data exchange with a server
  - In an asynchronous way (and also synchronous)
  - Without having to reload the whole page
- Corresponds to a set of technologies
  - HTML and CSS
  - DOM
  - XML or JSON (or other)
  - JavaScript, especially the XMLHttpRequest object

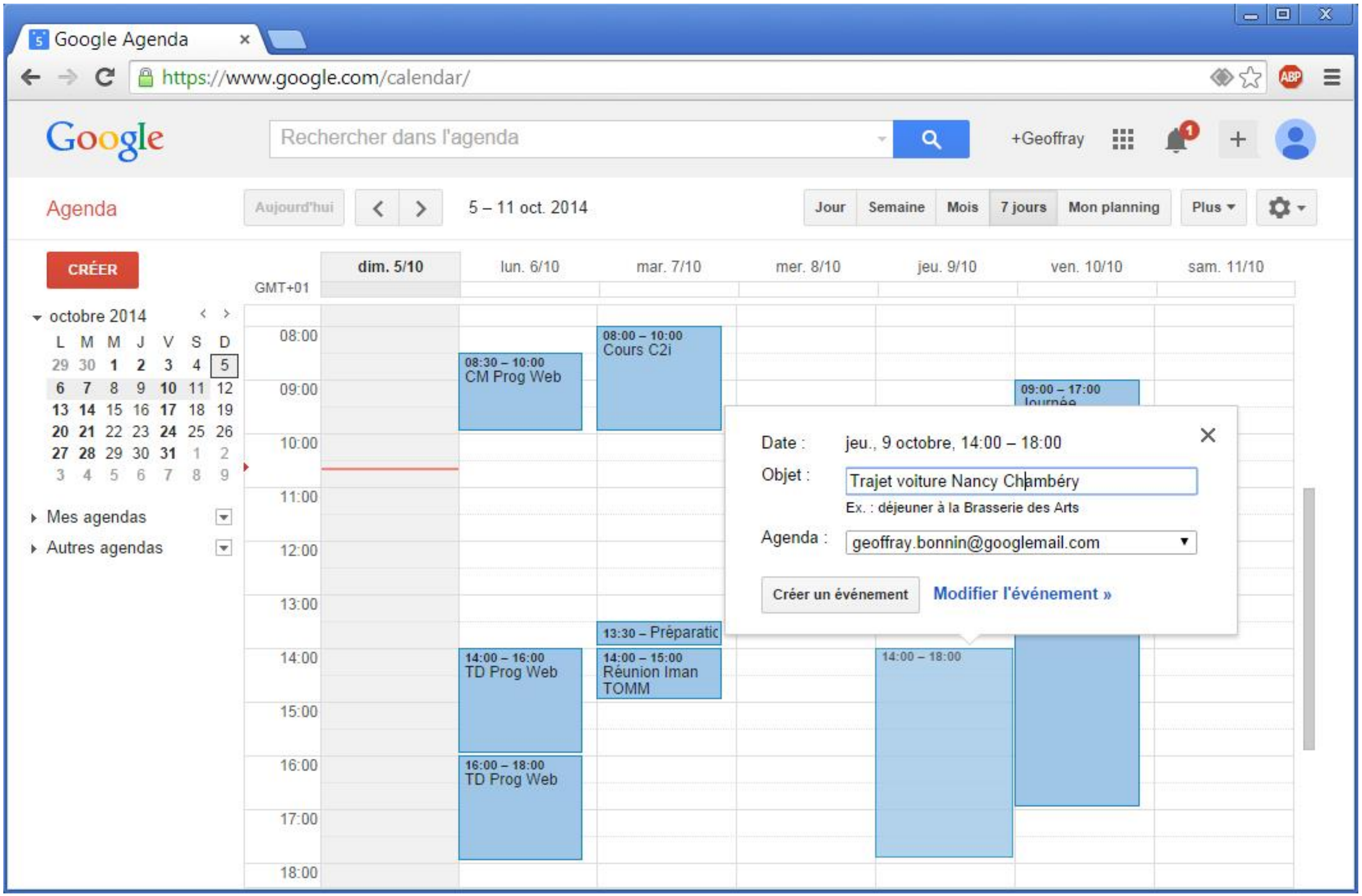
Google and Ajax

- Google Suggest (2004)



- Gmail (2005)
- Google Maps (2005)
- Google Reader, Google Calendar (2006)…

Focus on Google Calendar



08:30 – 10:00 CM Prog Web

08:00 – 10:00 Cours C2i

09:00 – 17:00 Journée

13:30 – Préparatic

14:00 – 16:00 TD Prog Web

14:00 – 15:00 Réunion Iman TOMM

16:00 – 18:00 TD Prog Web

14:00 – 18:00

Date : jeu., 9 octobre, 14:00 – 18:00

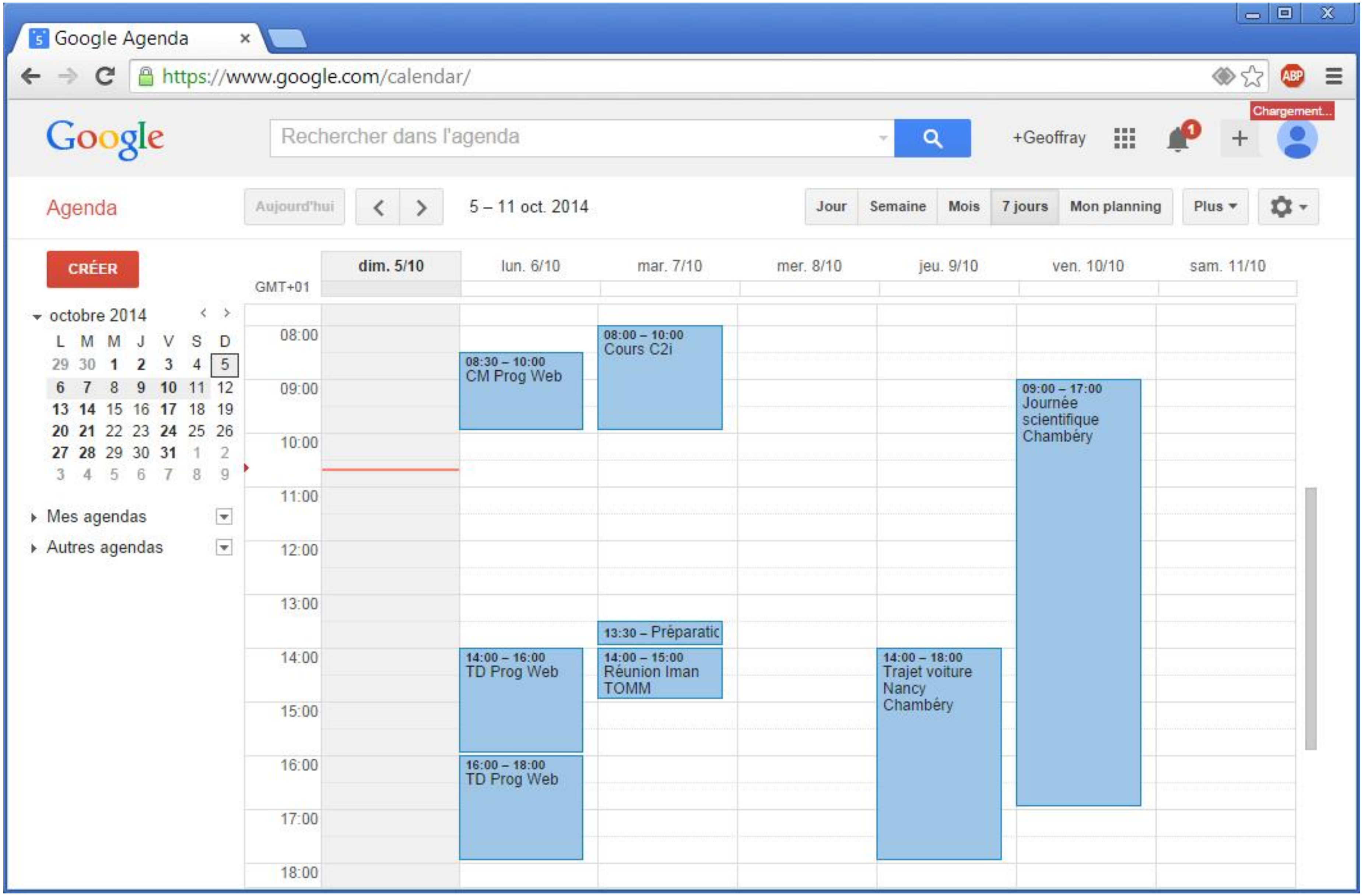
Objet : Trajet voiture Nancy Chambéry

Ex. : déjeuner à la Brasserie des Arts

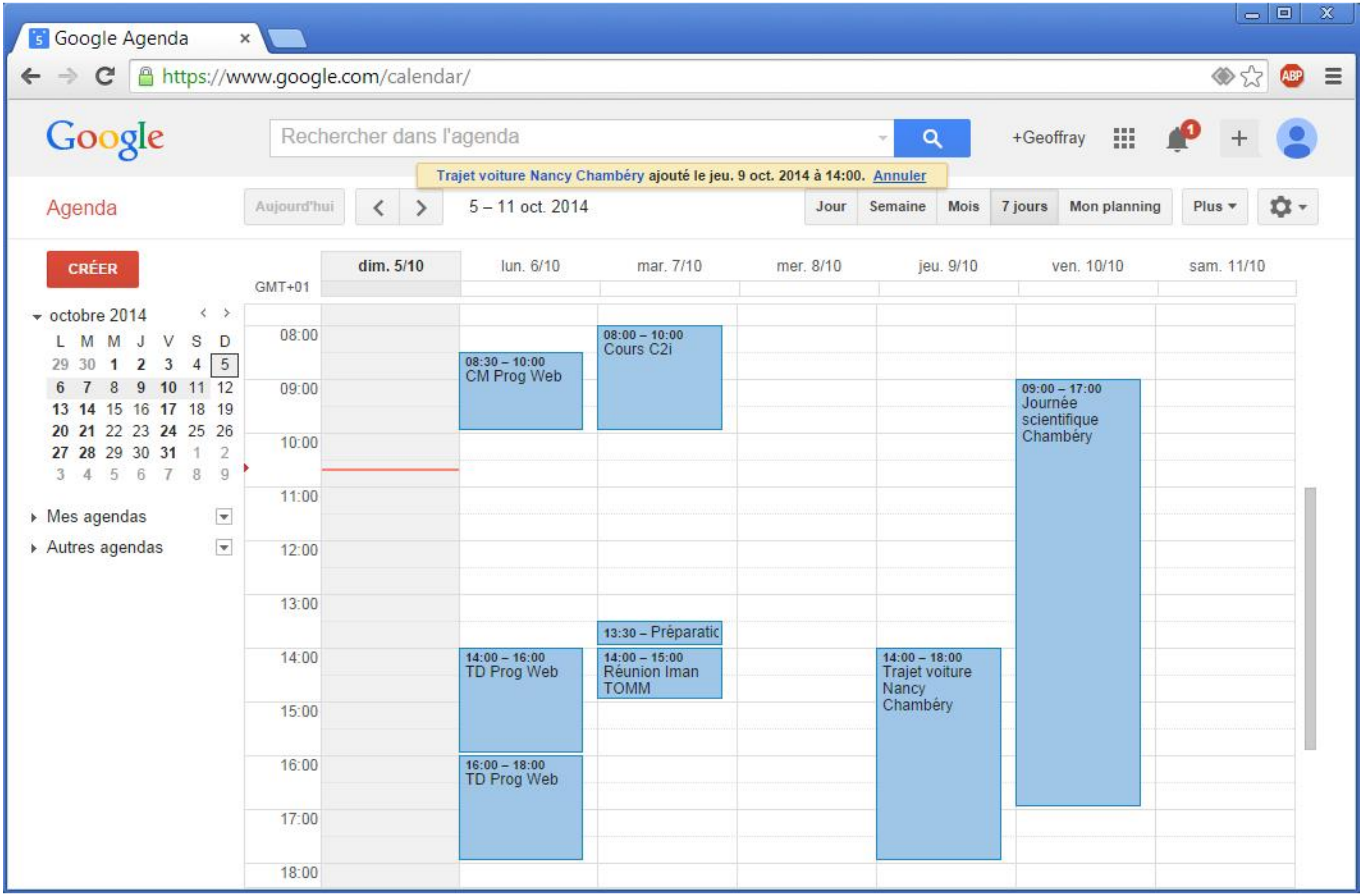
Agenda : geoffray.bonnin@googlemail.com

Créer un événement

Modifier l'événement »



dim. 5/10	lun. 6/10	mar. 7/10	mer. 8/10	jeu. 9/10	ven. 10/10	sam. 11/10
	08:30 – 10:00 CM Prog Web	08:00 – 10:00 Cours C2i			09:00 – 17:00 Journée scientifique Chambéry	
	14:00 – 16:00 TD Prog Web	13:30 – Préparatic		14:00 – 18:00 Trajet voiture Nancy Chambéry		
	16:00 – 18:00 TD Prog Web	14:00 – 15:00 Réunion Iman TOMM				



Trajet voiture Nancy Chambéry ajouté le jeu. 9 oct. 2014 à 14:00. Annuler

5 – 11 oct. 2014

octobre 2014

L	M	M	J	V	S	D
29	30	1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	31	1	2
3	4	5	6	7	8	9

dim. 5/10 · lun. 6/10 · mar. 7/10 · mer. 8/10 · jeu. 9/10 · ven. 10/10 · sam. 11/10

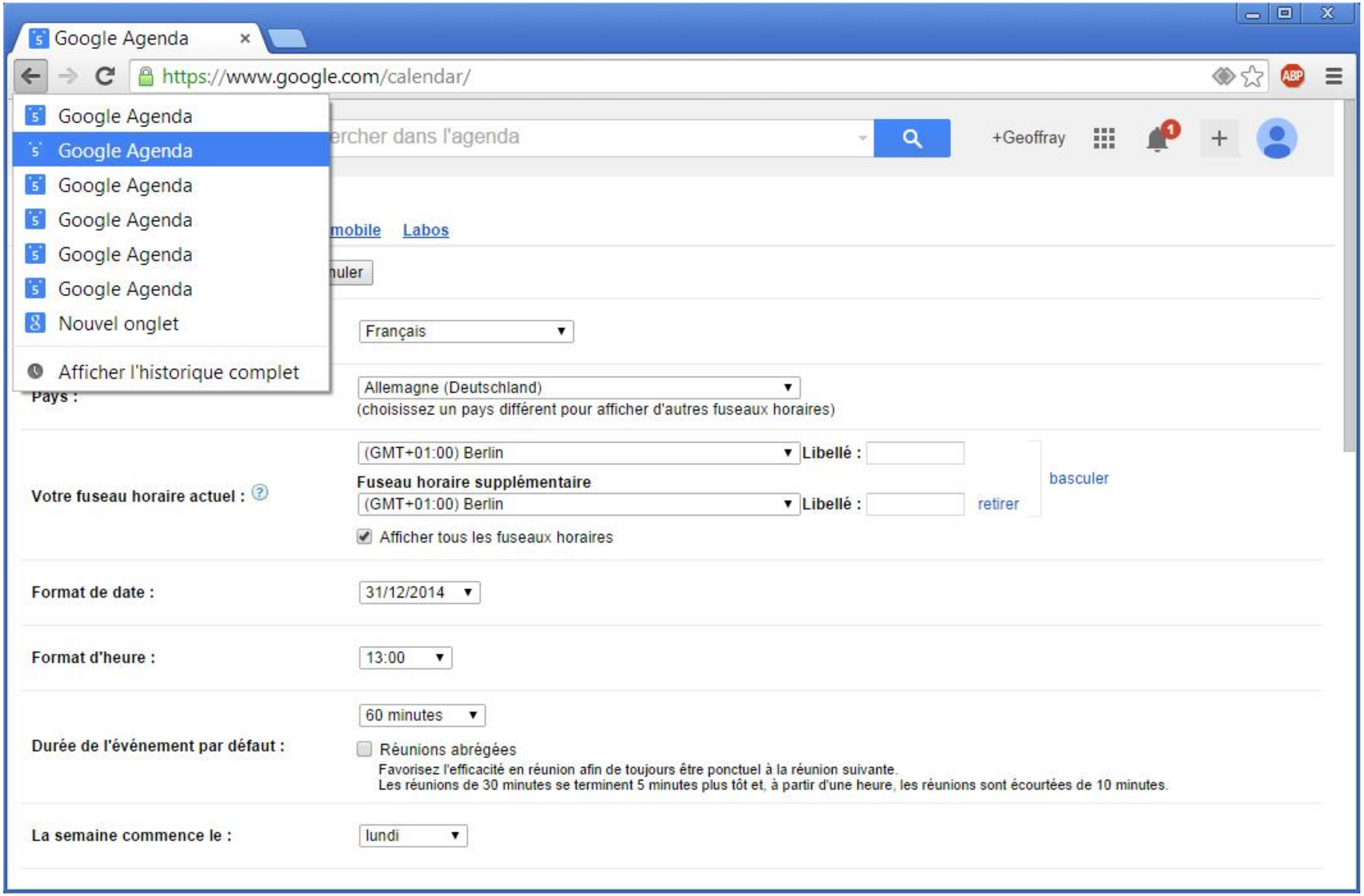
08:00 · 09:00 · 10:00 · 11:00 · 12:00 · 13:00 · 14:00 · 15:00 · 16:00 · 17:00 · 18:00

lun. 6/10:
08:30 – 10:00 CM Prog Web
14:00 – 16:00 TD Prog Web
16:00 – 18:00 TD Prog Web

mar. 7/10:
08:00 – 10:00 Cours C2i
13:30 – Préparatic
14:00 – 15:00 Réunion Iman TOMM

jeu. 9/10:
14:00 – 18:00 Trajet voiture Nancy Chambéry

ven. 10/10:
09:00 – 17:00 Journée scientifique Chambéry



Français

Pays : Allemagne (Deutschland)
(choisissez un pays différent pour afficher d'autres fuseaux horaires)

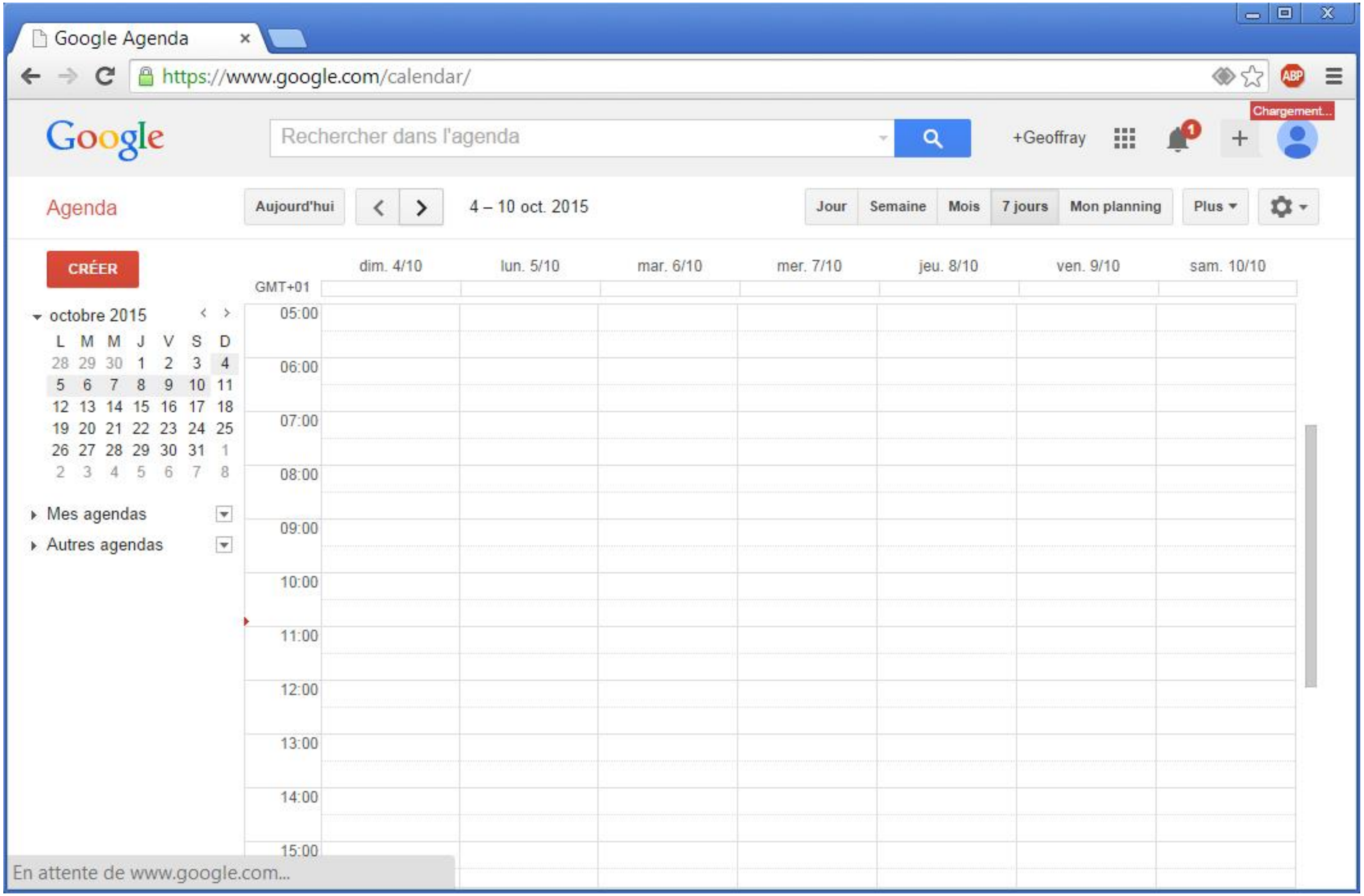
Votre fuseau horaire actuel : (GMT+01:00) Berlin **Libellé :**
Fuseau horaire supplémentaire
(GMT+01:00) Berlin **Libellé :** retirer
basculer
☑ Afficher tous les fuseaux horaires

Format de date : 31/12/2014

Format d'heure : 13:00

Durée de l'événement par défaut : 60 minutes
☐ Réunions abrégées
Favorisez l'efficacité en réunion afin de toujours être ponctuel à la réunion suivante.
Les réunions de 30 minutes se terminent 5 minutes plus tôt et, à partir d'une heure, les réunions sont écourtées de 10 minutes.

La semaine commence le : lundi

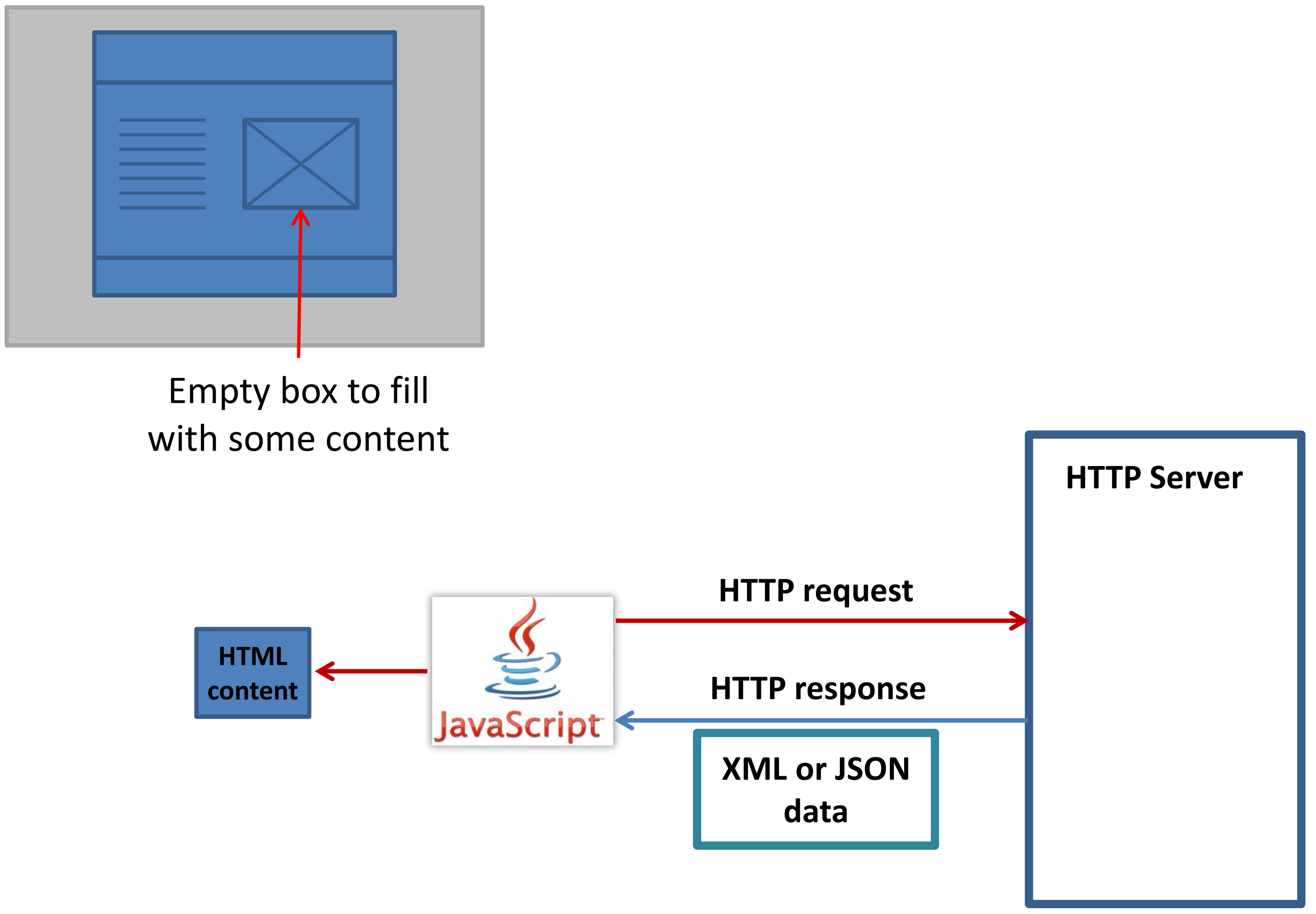


L	M	M	J	V	S	D
28	29	30	1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	1
2	3	4	5	6	7	8

Some rules for Ajax

1. Inform the user of what is happening
   everything is in a single page
2. Explicitly handle the back button
   No previous page
3. Clear distinction between content and design
   Transfer optimization
4. Cache data locally
   Transfer optimization

How does AJAX work?


Empty box to fill
with some content
HTTP Server
HTTP request
HTML
content
JavaScript
HTTP response
XML or JSON
data

The data file

- Contains data to be used in the page
- Must be on the same server
- Usually XML or JSON
  - XML

```
<data>
  <temperature>26</temperature>
  <humidite>70</humidite>
  <tendance>Variable</tendance>
</data>
```

  - JSON

```
{temperature:26, humidite:70, tendance:'Variable'}
```

Ajax call with JavaScript

- Uses the `XMLHttpRequest` object
  - Sends HTTP requests
  - Gets the responses
- Supported by (almost) all the current browsers
  - IE 7+, Firefox, Chrome, Safari and Opera

```
var req = new XMLHttpRequest();
```

  - IE 5 and 6: `ActiveXObject`

```
var eq = new ActiveXObject("Microsoft.XMLHTTP");
```

Sending an HTTP request

- Two methods have to be used
  - `open(`*`method, url, async`*`)`: specifies the type of request, the URL and mode of communication.
  - `send()`: Sends the requests.
- Example

```
var req = new XMLHttpRequest();
req.open('GET', 'data.json', true);
req.send();
```

Getting the response

Four important elements

- The `req.onreadystatechange` event
  When `req.readyState` changes
- The `req.readyState` property
  Values ranging between 0 and 4
  response has arrived: 4
- The response code: `req.status`
  200 : « OK », 404 : « Page not found », etc.
- The content of the response
  `req.responseText` or `req.responseXML`

```
function loadJSONDoc(){
   // Create XMLHttpRequest object (Check browser)
   var xmlhttp;

   if (window.XMLHttpRequest){
      // code for IE7+, Firefox, Chrome, Opera, Safari
      xmlhttp = new XMLHttpRequest();
   }
   else{
      // code for IE5 and IE6
      xmlhttp = new ActiveXObject("Microsoft.XMLHTTP");
   }

   // Things to do when a response arrives
   xmlhttp.onreadystatechange = function(){
      if (xmlhttp.readyState == 4 && xmlhttp.status == 200){
         // Change div content to the text content of the response
         document.getElementById("myDiv").innerHTML = xmlhttp.responseText;
      }
   }

    // Initialize request
   xmlhttp.open("GET", "data.json", true);

    // Send
   xmlhttp.send();
}
```

XML

Why XML?

Aim: data exchange

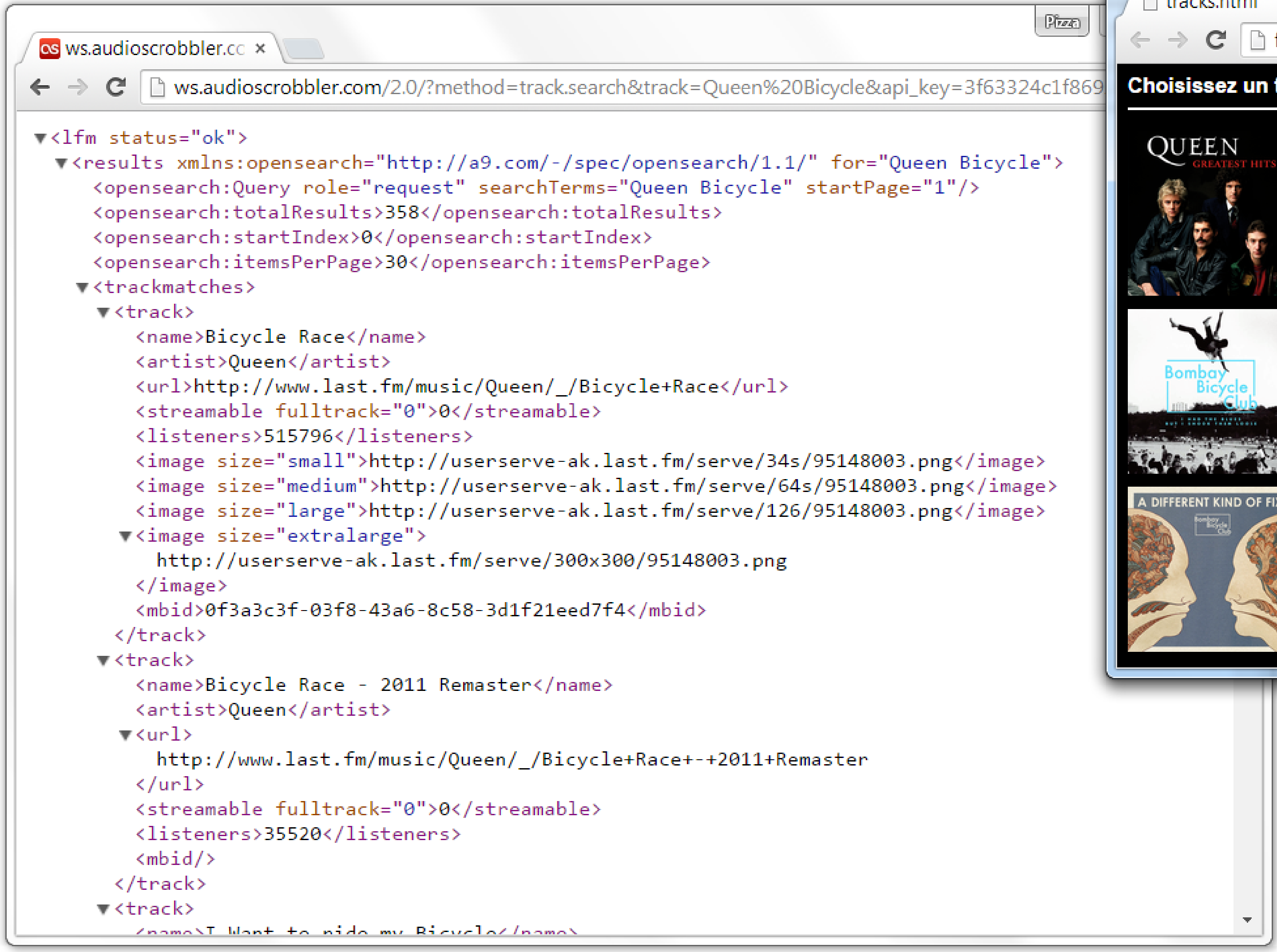
- Beginning of Internet: many text files with different formats

  artist:queen|title:bicycle race|year:1978

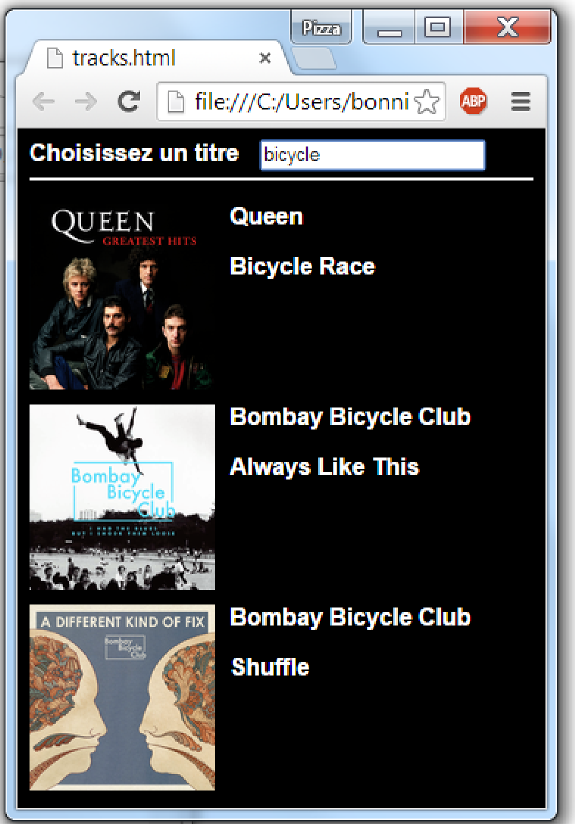
- Need of standards:

  XML 1.0 released in 1998 (20 year after Bicycle race)
  XML 1.1 released in 2004 (and nobody uses it)

Example of usage of XML


ws.audioscrobbler.cc
ws.audioscrobbler.com/2.0/?method=track.search&track=Queen%20Bicycle&api_key=3f63324c1f869

```
▼<lfm status="ok">
  ▼<results xmlns:opensearch="http://a9.com/-/spec/opensearch/1.1/" for="Queen Bicycle">
     <opensearch:Query role="request" searchTerms="Queen Bicycle" startPage="1"/>
     <opensearch:totalResults>358</opensearch:totalResults>
     <opensearch:startIndex>0</opensearch:startIndex>
     <opensearch:itemsPerPage>30</opensearch:itemsPerPage>
    ▼<trackmatches>
      ▼<track>
         <name>Bicycle Race</name>
         <artist>Queen</artist>
         <url>http://www.last.fm/music/Queen/_/Bicycle+Race</url>
         <streamable fulltrack="0">0</streamable>
         <listeners>515796</listeners>
         <image size="small">http://userserve-ak.last.fm/serve/34s/95148003.png</image>
         <image size="medium">http://userserve-ak.last.fm/serve/64s/95148003.png</image>
         <image size="large">http://userserve-ak.last.fm/serve/126/95148003.png</image>
        ▼<image size="extralarge">
           http://userserve-ak.last.fm/serve/300x300/95148003.png
         </image>
         <mbid>0f3a3c3f-03f8-43a6-8c58-3d1f21eed7f4</mbid>
       </track>
      ▼<track>
         <name>Bicycle Race - 2011 Remaster</name>
         <artist>Queen</artist>
        ▼<url>
           http://www.last.fm/music/Queen/_/Bicycle+Race+-+2011+Remaster
         </url>
         <streamable fulltrack="0">0</streamable>
         <listeners>35520</listeners>
         <mbid/>
       </track>
      ▼<track>
         <name>I Want to ride my Bicycle</name>
```



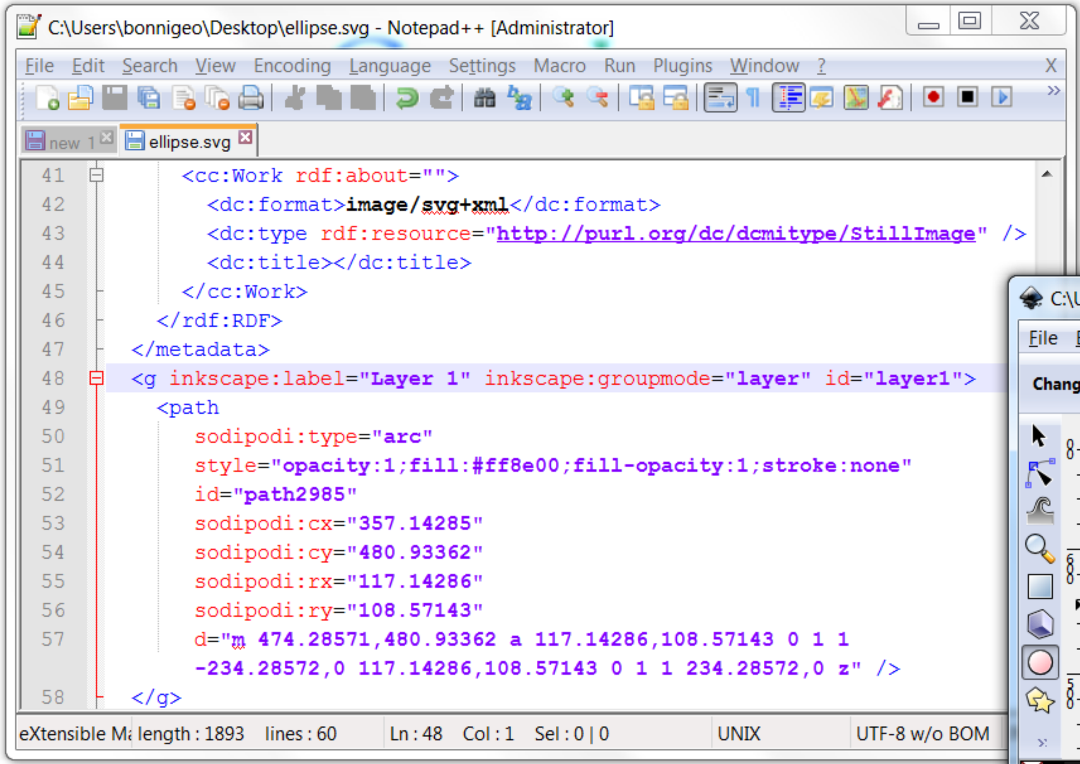
Choisissez un titre bicycle

Queen
Bicycle Race

Bombay Bicycle Club
Always Like This

Bombay Bicycle Club
Shuffle

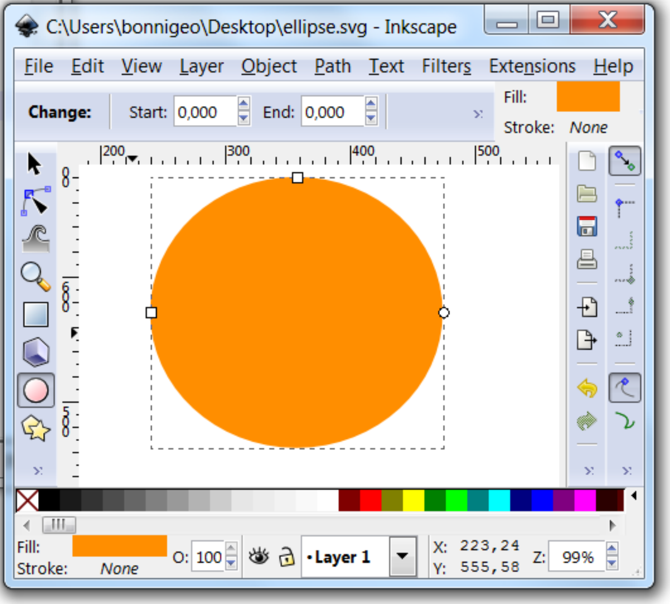
Another example


C:\Users\bonnigeo\Desktop\ellipse.svg - Notepad++ [Administrator]

File Edit Search View Encoding Language Settings Macro Run Plugins Window ?
new 1
ellipse.svg

```
      <cc:Work rdf:about="">
        <dc:format>image/svg+xml</dc:format>
        <dc:type rdf:resource="http://purl.org/dc/dcmitype/StillImage" />
        <dc:title></dc:title>
      </cc:Work>
    </rdf:RDF>
  </metadata>
  <g inkscape:label="Layer 1" inkscape:groupmode="layer" id="layer1">
    <path
       sodipodi:type="arc"
       style="opacity:1;fill:#ff8e00;fill-opacity:1;stroke:none"
       id="path2985"
       sodipodi:cx="357.14285"
       sodipodi:cy="480.93362"
       sodipodi:rx="117.14286"
       sodipodi:ry="108.57143"
       d="m 474.28571,480.93362 a 117.14286,108.57143 0 1 1
       -234.28572,0 117.14286,108.57143 0 1 1 234.28572,0 z" />
  </g>
```

eXtensible Ma length : 1893 lines : 60 Ln : 48 Col : 1 Sel : 0 | 0 UNIX UTF-8 w/o BOM


C:\Users\bonnigeo\Desktop\ellipse.svg - Inkscape

File Edit View Layer Object Path Text Filters Extensions Help
Change:
200
300
400
500

Start: 0,000 End: 0,000 Fill: Stroke: None

Fill: Stroke: None O: 100 Layer 1 X: 223,24 Y: 555,58 Z: 99%

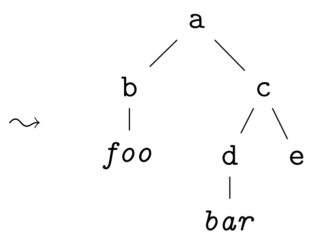
What is XML?

- Extensible Markup Language
  - Metalanguage based on tags
  - Is a W3C recommendation
- Allow
  - To represent data
  - Using a tree structure
- Does not allow to “do” something
  - Not a programming language
  - Not a network protocol
- Advantages
  - Very easy to learn
  - Vast amount of tools that exploits it

XML – Data model

- XML provides an encoding to build **trees**

```
<a>
  <b>foo</b>
  <c>
    <d>bar</d>
    <e/>
  </c>
</a>
```



- These trees have several types of **nodes**
  - **Element nodes** (here a, b, etc.): have a name and can have any number of children.
  - **Text nodes** (here foo, bar): have some text content and cannot have children.

Node types

12 type, including:

1. Each XML document is encapsulated in a **document node**. Exactly one of the children of this node must be an element node.
2. The **element nodes**. They can have element, processing instruction, comment, and text nodes as children.
3. Element nodes may own **attribute nodes**, which consist of a name and a value. Attribute names must be unique within one element.
4. The **Text nodes**.
5. The **namespace nodes**.
6. The **processing instruction nodes**: `<?target Content may be any string ?>`
7. The **comment nodes**: `<!-- This is a comment -->`
8. The **doctype nodes**: `<!DOCTYPE cours PUBLIC "cours.dtd">`
9. The **CDATA nodes** (character data): `<![CDATA[<p>Le XML c’est cool</p>]]>`

…

Example

```
<?xml version='1.0' encoding='utf-8'?>
<!-- Example from www.w3.org -->
<?xml-stylesheet type='text/xsl'?>
<catalog xmlns='http://www.example.com/catalog'
    xmlns:xlink='http://www.w3.org/1999/xlink'
    xmlns:html='http://www.w3.org/1999/xhtml'>
  <tshirt code='T1534017' sizes='M L XL'
    xlink:href='http://example.com/0,,1655091,00.html'>
    <title>Staind: Been Awhile Tee Black (1-sided)</title>
    <description>
      <html:p>
       Lyrics from the hit song 'It's Been Awhile' are shown in
       white, beneath the large 'Flock &amp; Weld' Staind logo.
      </html:p>
    </description>
    <price currency='EUR'>25.00</price>
  </tshirt>
</catalog>
```

Processing instruction node

Comment node

Namespace node

Element node

Text node

Attribute node

XML declaration

- Specific processing instruction node

```
<?xml version="1.0" encoding="utf-8" standalone="yes" ?>
```

- Attributes
  - Version: usually “1.0”
  - Encoding: “UTF-8” (or other)
  - Standalone:
    - “yes”: no DTD (grammar)
    - “no”: external DTD

CDATA section

```
<bio>
  <links>
    <link rel="original" href="http://www.last.fm/music/Bobby+McFerrin/+wiki"/>
  </links>
  <published>Fri, 19 Aug 2011 01:06:30 +0000</published>
  <summary>
    <![CDATA[
      Bobby McFerrin (born New York City, March 11, 1950) is a <a href="http://www.last.fm/tag/jazz" class="bbcode_tag"
      rel="tag">jazz</a>-influenced a cappella <a href="http://www.last.fm/tag/vocal" class="bbcode_tag"
      rel="tag">vocal</a> performer and conductor. A ten-time Grammy Award winner, he is one of the world's best known
      vocal innovators and improvisers. His song &quot;<a
      href="http://www.last.fm/music/Bobby+McFerrin/_/Don't+Worry%2C+Be+Happy">Don't Worry, Be Happy</a>&quot; (featured
      in the 1988 movie Cocktail, and the 2005 movie Jarhead) was a #1 U.S. pop hit in 1988. He has also worked in
      collaboration with instrumental performers including pianist <a href="http://www.last.fm/music/Chick+Corea"
      class="bbcode_artist">Chick Corea</a> and cellist <a href="http://www.last.fm/music/Yo-Yo+Ma"
      class="bbcode_artist">Yo-Yo Ma</a>. This collaboration has established him as an ambassador of both the <a
      href="http://www.last.fm/tag/classical" class="bbcode_tag" rel="tag">classical</a> and <a
      href="http://www.last.fm/tag/jazz" class="bbcode_tag" rel="tag">jazz</a> worlds. <a
      href="http://www.last.fm/music/Bobby+McFerrin">Read more about Bobby McFerrin on Last.fm</a>.
    ]]>
  </summary>
  <content>
    <![CDATA[
      Bobby McFerrin (born New York City, March 11, 1950) is a <a href="http://www.last.fm/tag/jazz" class="bbcode_tag"
      rel="tag">jazz</a>-influenced a cappella <a href="http://www.last.fm/tag/vocal" class="bbcode_tag"
      rel="tag">vocal</a> performer and conductor. A ten-time Grammy Award winner, he is one of the world's best known
      vocal innovators and improvisers. His song &quot;<a
      href="http://www.last.fm/music/Bobby+McFerrin/_/Don't+Worry%2C+Be+Happy">Don't Worry, Be Happy</a>&quot; (featured
      in the 1988 movie Cocktail, and the 2005 movie Jarhead) was a #1 U.S. pop hit in 1988. He has also worked in
      collaboration with instrumental performers including pianist <a href="http://www.last.fm/music/Chick+Corea"
      class="bbcode_artist">Chick Corea</a> and cellist <a href="http://www.last.fm/music/Yo-Yo+Ma"
      class="bbcode_artist">Yo-Yo Ma</a>. This collaboration has established him as an ambassador of both the <a
      href="http://www.last.fm/tag/classical" class="bbcode_tag" rel="tag">classical</a> and <a
      href="http://www.last.fm/tag/jazz" class="bbcode_tag" rel="tag">jazz</a> worlds. <a
      href="http://www.last.fm/music/Bobby+McFerrin">Read more about Bobby McFerrin on Last.fm</a>. User-contributed text
      is available under the Creative Commons By-SA License and may also be available under the GNU FDL.
```

Element node or attribute node?

```
<personne guitare1="papalardo"  guitare2="gibson"/>
```

or

```
<personne>
  <guitare>papalardo</guitare>
  <guitare>gibson</guitare>
</personne>
```

?

– If value not much repeated → not important

– Else → element node

Constraints of XML

- Well formed documents
  - Every opened tags must be closed.
  - The elements can be combined but must not overlap (e.g., ~~<div><p></div></p>~~)
  - There must be only one root element (e.g., <html>)
  - All attributes must have double quotes
  - An element must not have two attributes with the same name.
  - Comment and processing instruction nodes are not allowed inside of tags.
  - No unescaped character ‘<’ or ‘&’ in text and attribute nodes.
  - The name of an element cannot start with a number.
  - …
- To check an XML document is well formed
  - Simple solution: open it with a web browser
  - Better: use an XML parser (Xerces, MSXML, Expat, libxml…)

Naming conventions

- Lower case letters for attributes and element names
- Avoid special characters for attributes and element names
- Name composed of several words: ‘-’, ‘_’ or CamelCase

```
<value-of/>
<value_of/>
<valueOf/>
```

JSON

JSON

- JavaScript Object Notation
- Derived from JavaScript
- Similarities with XML
  - Plain text
  - Kind of tree structure
- Differences with XML
  - Shorter (no closing tags)
  - No need to have a dedicated parser when used by JavaScript
  - Allows to include arrays

Example

```
{
  "firstName": "John",
  "lastName": "Smith",
  "age": 25,
  "address": {
    "streetAddress": "21 2nd Street",
    "city": "New York",
    "state": "NY",
    "postalCode": "10021"
  },
  "phoneNumber": [ { "type": "home", "number": "212 555-1239" },
                   { "type": "fax", "number": "646 555-4567" } ],
  "gender": { "type": "male" }
}
```

JSON → JavaScript object

```
<body>
  <p id="artist"></p>

  <script>
    var text = '{"name": "Jimi", "instrument": "Guitar"}'
    var obj = JSON.parse(text);
    var pArtist = document.getElementById("artist");
    pArtist.innerHTML = obj.name + "<br/>" + obj.instrument;
  </script>
</body>
```

XML or JSON?

- Advantages of XML over JSON
  - Many technologies exist that exploits it
  - Grammars and schemas
- Advantages of JSON over XML
  - Lower space usage, faster
  - Easier for web development (JavaScript)