MODERN WEB 2.0 TOOLS

FOR EDUCATIONAL ASTRONOMY

Davide Ricci[1,2], Luciano Nicastro[1], Miguel Ángel Pio[3]

On behalf of the GLORIA collaboration

[1]INAF/Istituto di Astrofisica Spaziale e Fisica Cosmica, Via Gobetti 101, Bologna, (ITALY),

[2]Author Instituto de Astronomía, UNAM, AP 877, Ensenada, B.C. 22800 (MEXICO),

[3]Instituto de Astrofísica de Canarias (SPAIN)

indy@astrosen.unam.mx, nicastro@iasfbo.inaf.it, mpio@iac.es

Abstract

In the framework of the GLORIA (GLObal Robotic-telescopes Intelligent Array) project, we present an overview of what we believe are the best web technologies that can be applied to build browser-based educational tools. A step-by-step code tutorial will help the reader to understand how this result can be achieved. We also present these concepts in two websites for educational purposes. The first one uses real images of the Sun taken during the last transit of Venus (June 2012). Students can use these images to calculate the Sun-Earth distance. The second one makes use of real meteorological data taken in Australia during the last total solar eclipse (November 2012) to interactively calculate the time lag between observed changes in luminosity and ambient temperature.

Keywords: Astronomy, Web 2.0, HTML5, CSS3, jQuery, d3.js, PHP, mysqli, MySQL

1 INTRODUCTION

Young students spend a lot of time in front of a computer for leisure (offline or online gaming, social networking, etc.) but also for educational purposes (homeworks, research). For this reason it is common habit to refer to this generation with the term “digital natives”. These “modern” students are not simply computer users, but experienced internet users who use web browsers as the main interface for self learning and surfing the internet in general. As such they have developed a “taste” for web design that goes beyond the pure concept of graphical presentation of the content. It is simpler and faster to find or learn something on the web if the site is user friendly, browsable in an intuitive way and if it implements what we now consider as “natural” actions while surfing. For example, if we use the mouse wheel on a standard web page or a text, we expect to scroll it; if we use the same wheel on an image, we expect to zoom it.

It is interesting and important to take advantage of this common knowledge of the latest internet technologies to build modern web-based scientific educational tools, in particular in scientific fields like astronomy. In fact astronomy is a particularly popular science among young students and the general public: televisions, websites and newspapers regularly inform people about exoplanets discoveries, Mars rover results, spectacular nebulae images, and much more.

But what happens when the students try to increase these knowledge or deepen these information through the Internet for homework, school laboratories or simply for personal interest? Except for some cases, they probably land on old-style sites or static applications which are completely outdated with respect to their expectations. This happens for several reasons: first of all scientists, and astronomers in particular, concentrate their efforts on scientific communication (mainly papers and oral presentations) and their technical knowledge is linked to the related instruments to produce them, e.g. LaTeX and PowerPoint. Moreover, when they decide to show their results on a website, they usually adopt the same (typically static) schema of the scientific communications. However, a website is not a

paper, nor an oral presentation. A reader expects interactivity, usability, a text easy to read and a consistent layout: all things that we consider “natural”, as explained above, while surfing professional sites.

Modern web technologies can help not professional web-designers, like astronomers, to achieve the goal to effectively explain their complex work to students and the general public and to spread news about their discoveries via educational sites. Moreover web based citizen science projects have proven particularly successful when clear and intuitive user interfaces are implemented. This is for example the case of the projects proposed by zooniverse.org and we hope it will be also true for the GLORIA project (see gloria-project.eu for all the details). GLORIA is an EC funded project with the aim to give free and open-access to a network of robotic telescopes worldwide. Moreover a web 2.0 environment will allow users to do research in astronomy by observing with such telescopes and/or by analysing data that other users have acquired. The use of effective web technologies is of great importance to manage and make available to the users images and data collected by a wide variety of instruments. In particular simple and efficient graphics tool together with a distributed database system will be required.

We present such technologies in Sect. 2. In Sect. 3 we show them in a simple step-by-step tutorial. We also present some of them in action in two educational web-based tools (Sects. 4.1 and 4.2). In Sect. 5 we draw the conclusions.

2 TECHNOLOGIES

The web is constantly changing and it is important for the developers to be updated to all the latest standards and technologies. In this section we present an overview of the basic instruments that could be used to build modern web tools for educational astronomy.

2.1 HTML5

Web pages are written in html, a markup language such as LaTeX which is interpreted by the browser to show the content as we are used to see. The last revision of this language, called html5, is foreseen to be released on July 2014 and offers several new features[1], most of which are already supported by the major browsers. In particular:

input types: html5 provides a set of new attributes for input forms such as `datetime`, `datetime-local`, `number`, `range`, `url`, `email` that could help to check client-side the input values for a query in a database. Moreover, several new touch devices (such as tablets) use these attributes to show to the user the more suitable keyboard layout (alphabetical, numerical, custom-defined);

form attributes: new attributes such as `step` (defining the allowed intervals of an input field), `required`, and `pattern` (which uses regular expressions) could help to avoid errors in the submission of a form in the context, e.g. astronomy, where the nomenclature is sometimes very complex;

semantic tags: the `<article>`, `<section>`, `<figure>`, `<figcaption>`, `<summary>` and `<details>` can help astronomers to present their work following the scheme already used for their scientific publications, simplifying their work and helping to design consistent web layouts;

MathML: this technology provides an xml-based markup, already supported by the major browsers (excepted Chromium) to write high quality mathematical formulae. The markup is very heavy, butseveral converters from/to the LaTeX syntax are already available on the Internet[2];

media tags: the `<audio>`, `<video>` and `<track>` will provide a valid cross-browser alternative to flash-based plugins for online access to multimedia contents like tutorials and video samples;

geolocation: used to report user’s position with a simple click. It can be used for survey statistics;

application cache and local storage: they offer an opportunity to create an offline version of a web application. they can be used to reduce the server load because the browser will only download updated or changed resources from the server; local storage also overrides the limitations of the cookies in storing local information.

1 http://www.w3schools.com/html5/default.asp
2 http://latex2mathml.freewebmaster.fr/demo.php

server-sent events: this technology provides the possibility to obtain real-time events as emitted by the server without a specific request from the client (e.g. with a click, a mouse movement, a polling or a WebSocket). It is a kind of communication that reminds the old concept of television;

canvas: this element allows to draw graphics and display data, on the fly, on a web page. It can be used for dynamic plots, for displaying FITS images (see fits.gsfc.nasa.gov) or, for example, to superpose to the image additional layers with labels and marks;

SVG: inline vector graphics management can be used for plots and data representation. Simple or complex vector shapes can be directly included between the `<svg></svg>` tags using a syntax similar to html and xml, and it can be seen either as an alternative or complementary to the `<canvas>` element;

WebGL: although it is not an html5 element, it can be seen as a side technology of it. It allows the rendering of interactive 3D graphics without the use of browser plug-ins via the `<canvas>` element. WebGL is a low-level technology and requires good programming skills.

It is crucial to understand that the only thing requested to the user to be able to work with all these kind of technologies is nothing but an *updated browser*.

2.2 CSS3

In modern web design, it is essential to separate the content (written in html) from the layout (defined by the css stylesheet[3]). It is the same concept that researchers apply every day while using LaTeX, where the layout and graphical presentation is defined by the style sheet provided by the publisher and is normally described in a `.sty` file.
The improvements coming with the third version of the css language, or css3, are still in development. The new modules of css3 help to build attractive pages by introducing new methods for backgrounds, borders, effects and animations. css3 can also help to adapt the layout of a web application for different kind of devices.

2.3 jQuery and d3.js

Normally, in web pages, client-side interaction is provided by JavaScript (JS). This language is not to be confused with java, a software platform based on a virtual machine which runs programs written in its specific object-oriented programming language, also named java. Several JS libraries, for any kind of purpose, are available through the Internet. In the last years the jquery library (jquery.com) grew in popularity due to its simplified way to access the html elements (using a syntax similar to that of the css selectors) and to manage asynchronous requests. The functions are chainable, cross-browser, and easily expandable with custom plugins. The popularity of jquery is currently very wide[4], The internet offers a long list of free resources that allow a developer to perform a great variety of tasks. We consider this library very suitable for educational web-based applications. It is simple enough to accomplish the most common operations required by a dynamic website, and generic enough to be adapted to our aims.

d3.js is a JS library for manipulating documents based on data (d3js.org), using in particular html and `<svg>` elements. Its syntax is very similar to that of jquery. The main concept is to use the html and `<svg>` elements to draw graphics on the web page [2]. Due to its flexibility, we foresee that the d3.js library will grow in popularity in the next months and should be considered one of the most promising libraries for real-time plotting and charting. An educational set of examples is provided by the author Mike Bostock (http://bl.ocks.org/mbostock) and daily updated. The library provides a set of methods for plot and charts (pie chart, scatter plot, bar chart, area and line chart, etc.), and a set of cartographic projections (Equirectangular, Mollewide, Aitoff, Orthographic) which are particularly useful to mark astronomical data points on the celestial sphere.

3 TUTORIAL

In order to show the potential of these technologies, we provide a short code tutorial aimed at building a simple interactive chart of bright stars (see Fig. 1). To run this code, it is necessary to have set a web

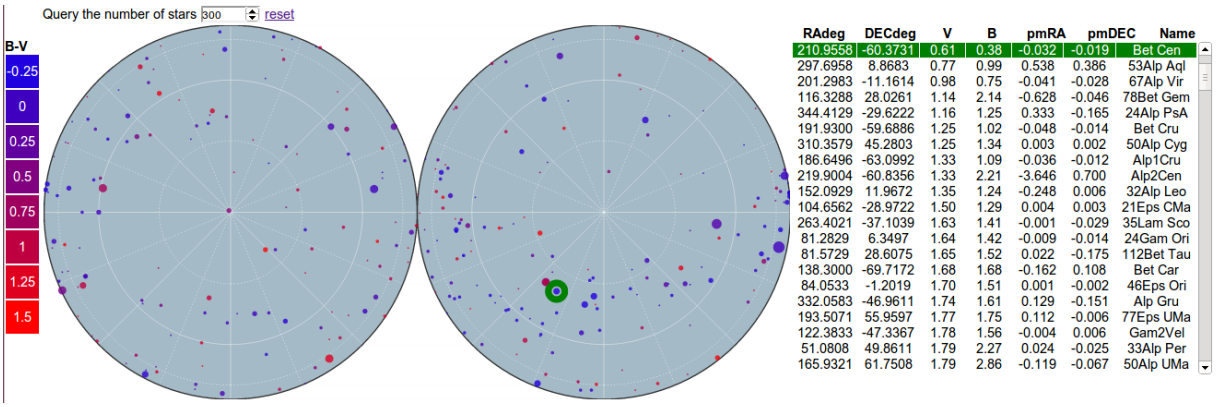
[3] http://www.w3schools.com/css/default.asp

[4] http://w3techs.com/technologies/details/js-jquery/all/all

```
<!doctype html>
<title>bright stars catalog</title>
<meta charset="utf-8">
<!-- <link rel="stylesheet" href="bright-stars.css"> -->
<style>
 body{          margin: 0px auto; font-family: Helvetica, sans-serif; }
 h1{            padding: 30px; font-weight: 300;                }
 svg{           border: 1px solid white; float: left;           }
 circle{        fill-opacity: 0.8;                          }
 circle.mouseover{ stroke-width: 20; stroke: green;             }
 tr.mouseover, td.mouseover{    background-color: green; color: white;  }
 td, th{        width: 110px; text-align: center;               }
 th:last-child{   width: 120px;                              }
 thead, thead tr{  display: block;                           }
 tbody{         display: block; height: 400px; overflow: auto;      }
 .background {    fill: #a4bac7;                             }
 .foreground {    fill: none; stroke: #333; stroke-width: 1.5px;      }
 .graticule {     fill: none; stroke: #fff; stroke-width: .5px;       }
 .graticule :nth-child(2n) { stroke-dasharray: 2,2;              }
 :invalid{background: pink}
 div{ float: left; width:45px; margin-top:40px; }
 div>svg{cursor: pointer}
</style>
<script src="js/jquery.min.js"></script>
<script src="js/d3.v3.min.js"></script>
<script src="js/d3.geo.projection.v0.min.js"></script>
<body>
 <h1>bright stars catalog</h1>
 <label for="nstars">Query the number of stars</label>
 <input id="nstars" type="number" value="300" min="100" max="9000" step="100">
 <a href="./">reset</a>
 <div><strong>B-V</strong><!-- here d3.js will display the legend --></div>
 <article> <!-- here d3.js will display the chart --> </article>
 <aside>  <!-- here d3.js will display the table --> </aside>
<script src="bright-stars.4.js"></script>
<?php
// mysql connection (host, username, password, database)
$c = new mysqli('ross.iasfbo.inaf.it','generic','password','test');
$table = 'simpleBSC';
$order = ' order by '.$_GET['ord'];
$limit = ' limit '.$_GET['lim'];
$bvmin = $_GET['bvmin'];
$bvmax = $_GET['bvmax'];
$where = ' where (B-V)>'.$bvmin.' and (B-V)<'.$bvmax;
$result=$c->query("select * from ".$table .' limit 0');
$mt=$result->fetch_fields();
for($m=0;$m< count($mt);$m++){
$v=get_object_vars($mt[$m]);
$mt2[]=$v['name'];
}
$q= 'SELECT '.implode(',',$mt2).' from '.$table .$where .$order .$limit
$result=$c->query($q);
while($r = $result->fetch_assoc()) {
$r2[]=$r;
}
echo json_encode($r2);
?>
```

Listing 1. An example of HTML code containing elements of HTML5. It calls JS and CSS files.

server with php and the mysqli module necessary to interact with a remote database. We will introduce the html5 new input type number, the new data type, some semantic tags. We also show the usage of the mysqli extension of the php language which we use to connect to a MySQL database. We finally introduce jquery and several components of the new d3.js library to draw svg and html elements using JS.



RAdeg	DECdeg	V	B	pmRA	pmDEC	Name
210.9558	-60.3731	0.61	0.38	-0.032	-0.019	Bet Cen
297.6958	8.8683	0.77	0.99	0.538	0.386	53Alp Aql
201.2983	-11.1614	0.98	0.75	-0.041	-0.028	67Alp Vir
116.3288	28.0261	1.14	2.14	-0.628	-0.046	78Bet Gem
344.4129	-29.6222	1.16	1.25	0.333	-0.165	24Alp PsA
191.9300	-59.6886	1.25	1.02	-0.048	-0.014	Bet Cru
310.3579	45.2803	1.25	1.34	0.003	0.002	50Alp Cyg
186.6496	-63.0992	1.33	1.09	-0.036	-0.012	Alp1Cru
219.9004	-60.8356	1.33	2.21	-3.646	0.700	Alp2Cen
152.0929	11.9672	1.35	1.24	-0.248	0.006	32Alp Leo
104.6562	-28.9722	1.50	1.29	0.004	0.003	21Eps CMa
263.4021	-37.1039	1.63	1.41	-0.001	-0.029	35Lam Sco
81.2829	6.3497	1.64	1.42	-0.009	-0.014	24Gam Ori
81.5729	28.6075	1.65	1.52	0.022	-0.175	112Bet Tau
138.3000	-69.7172	1.68	1.68	-0.162	0.108	Bet Car
84.0533	-1.2019	1.70	1.51	0.001	-0.002	46Eps Ori
332.0583	-46.9611	1.74	1.61	0.129	-0.151	Alp Gru
193.5071	55.9597	1.77	1.75	0.112	-0.006	77Eps UMa
122.3833	-47.3367	1.78	1.56	-0.004	0.006	Gam2Vel
51.0808	49.8611	1.79	2.27	0.024	-0.025	33Alp Per
165.9321	61.7508	1.79	2.86	-0.119	-0.067	50Alp UMa

Figure 1. Interactive sky chart of the bright stars catalogue.

html5 and css3 get programmers used to a minimalist code. See Listing 1: the first three lines set the html doctype, the title and the character encoding, whereas the commented fourth line can be used to link an external css stylesheet. For the sake of simplicity, in this tutorial we embedded the style between the <style></style> tags, then we linked the external JS libraries mentioned in Sect. 2: jquery and d3.js (plus a d3.js plugin). The following <body> tag sets the point where the content displayed by the browser starts. This is already a fully valid html5 file.

The <input type="number"> tag, new in html5, sets a minimum, maximum and a step value to manage the desired number of stars. Then, two <article> and <aside> semantic tags of html5, are meant to embed a self consistent and a related content of a webpage, respectively, will contain the bright stars chart and a related table with coordinates, magnitudes and proper motions.

Finally, the last line will call the JS (d3.js) code to draw the chart and the table. Before explaining how to draw the chart and populate the table, we show in Listing 2 a simple php code that uses the mysqli extension to connect to a remote database located in Bologna: a new connection to the database is created, then the name of the database table is declared.

The following GET requests read the user defined parameters ord and lim, which define the column to use for ordering the table and the maximum number of entries to retrieve from the database, plus a filter for the limits in the color index. The first value is set in the JS file, the second one is set in the <input> tag of the html file. A range of 0.25 in color index can be selected by clicking on the legend. Then, the database is queried and the fields are fetched. An associative array which contains these fields is prepared by a for cycle and the second query provides the result, which populates the array using a while statement.

The results are finally encoded in the json (json.org) format and are used by the JS. The code can be tested directly by opening the php file in the browser. For example, assuming that the file is placed in the www-path directory of the local webserver:

http://localhost/www-path/bright-stars.php?lim=2&ord=V&bvmin=0.5&bvmax=1.0

The previous “page” address will output this json result:

```
[{"RAdeg":"219.8996","DECdeg":"-60.8353","V":"-0.01","B":"0.70","pmRA":"-3.642","pmDEC":"0.699",
        "Name":"Alp1Cen"},
 {"RAdeg":"79.1725","DECdeg":"45.9981","V":"0.08","B":"0.88","pmRA":"0.076","pmDEC":"-0.425",
        "Name":"13Alp Aur"}]
```

Once the html page is prepared and the php page is able to output the results, we show how to draw the chart by coding the JS file called in the last line of Listing 1. We use the library d3.js. First of all, we create the two orthographic projections for the northern and southern emisphere, using a method already implemented in d3.js:

```
var width = 900,
height = 450; // dimensions of the svg image

var projection = d3.geo.interrupt(d3.geo.orthographic.raw)
   .lobes([[ // northern hemisphere
      [[-180, 0], [-90,  90], [0, 0]], [[0, 0], [ 90,  90], [180, 0]]
     ], [ // southern hemisphere
      .rotate([90, 0, 90]) // we center it at poles
      .scale(height/2)
      .translate([width / 2, height / 2])
      .precision(.1);

var path = d3.geo.path().projection(projection);
```

```
    [[-180, 0], [-90, -90], [0, 0]], [[0, 0], [ 90, -90], [180, 0]]
   ]])
```

and we append the svg tag and the graticules inside the <article> tag:

```
var svg = d3.select("article").append("svg")
    .attr("width", width)
    .attr("height", height);
var defs = svg.append("defs");

defs.append("path")
    .datum({type: "Sphere"})
    .attr("id", "sphere")
    .attr("d", path);

defs.append("clipPath")
    .attr("id", "clip")
    .append("use")
    .attr("xlink:href", "#sphere");

svg.append("use")
    .attr("class", "background")
    .attr("xlink:href", "#sphere");
var graticule = d3.geo.graticule()
    .extent([[-180, -90], [180, 90]]);

svg.append("g")
   .attr("class", "graticule")
   .selectAll("path")
   .data(graticule.lines).enter()
   .append("path").attr("d", path);

svg.append("use")
    .attr("class", "foreground")
    .attr("xlink:href", "#sphere");
```

The table that will contain the results inside the <aside> tag is created as follows:

```
var table = d3.select("aside").append("table"),
   thead = table.append("thead"),
   tbody = table.append("tbody");
```

We define the color range for the B−V color index and we create the clickable legend::

```
var color = d3.scale.linear()
    .domain([-0.5,1.5]) // min and max color domain
    .range(['rgb(0,0,255)', 'rgb(255,0,0)']);

var q=40;
var step=0.125;
var legdata=[-0.25, 0.00, 0.25, 0.50, 0.75, 1.00, 1.25,
1.50];
var legend =d3.select('div').selectAll('svg')
    .data(legdata).enter()
    .append('svg') // one little svg square for each value
    .attr('height',q)
    .attr('width',q)
    .style('background-color',function(d){
       return color(d)})
    .on('mouseover', mouserange) // interaction with m
ouse events
    .on('mouseout', mouseout)

legend.append('text')
    .text(function (d){return d})
    .attr('x',20) // or we can define it in the css styleshe
et
    .attr('y',26)
    .attr('text-anchor','middle')
    .attr('fill','white')
    .attr('stroke','none');
```

Note that we defined two interactions with mouse events. The functions that handle these behaviors are shown at the end of the script. At start we define the default parameters used by the php code ($limit, $order, $bvmin, $bvmax), whereas we call the main function table2 on load, on changing the <input> tag value (retrieved using jquery) and on clicking on the legend:

```
var ord="V";
var bvmin0=d3.extent(legdata)[0];
var bvmax0=d3.extent(legdata)[1];
var bvmin=bvmin0;
var bvmax=bvmax0;
// on load
table2(
  $('[type="number"]').val(), ord, bvmin0, bvmax0
);
// on click on the legend
d3.selectAll('div>svg').on('click',function(d,i){
    bvmin=d-step;
    bvmax=d+step;
    console.log(bvmin, bvmax)
    table2($('[type="number"]').val(), ord, bvmin, bvmax)
})

// on number of stars select
d3.select('[type="number"]').on('change',function(){
    table2($(this).val(), ord, bvmin, bvmax)
});
```

The main function table2 basically calls the php file with the parameters previously defined, and it expects a json file as a result:

```
function table2(lim,ord,bmin,bmax){d3.json("./bright-stars.php?
lim="+lim+"&ord="+ord+"&bvmin="+bvmin+"&bvmax="+bvmax, function(error, data){
```

We define a scale range for the radius of the circles that will represent the stars. A linear scale is adopted for the brightness:

```
var radius = d3.scale.linear()
   .domain(d3.extent(data, function(d){ return d.V; }))
   .range([5.0,1.0]);
```

and we append these circles to the svg image. Note that these circles do not exist yet:

```
var circle = svg.selectAll("circle")
   .data(data);
```

Then the circles are entered in the drawing, and their x and y pixel position is calculated by projection, rounded, and placed on the celestial sphere:

```
circle.enter().append("circle")
   .attr('data-index',function(d,i){return i});

circle.attr("cx", function(d){ return projection([d.RAdeg,d.DECdeg])[0].toFixed(0) })
     .attr("cy", function(d){ return projection([d.RAdeg,d.DECdeg])[1].toFixed(0) })
     .attr("fill", function(d){return color((d.B-d.V).toFixed(2))})
     .attr("r", function(d){ return radius(d.V).toFixed(0) })// radius depends on V mag
     .on('mouseover', mousescroll)
     .on('mouseout', mouseout);
```

Here again we added a simple interaction while the mouse is over. We will define these interactions in two functions at the end of the script. When another query to the database is performed, the unused nodes are removed:

```
circle.exit().remove();
```

The same approach applies to the side table:

```
// column names from first data row
var columns = [];
for (var key in data[0]){ columns.push(key); }
```

The same way as the circle elements, the header row is appended, the data are bind, the rows are entered:

```
// appending the header row
var th = thead.selectAll("th")
   .data(columns).enter()
   .append("th")
   .attr('data-key',function(c){return c})
   .text(function(c){ return c });
```

Then a row for each object in the returned data is created together with a mouse interaction similar to the previous one (while on a row with the cursor, the corresponding star on the sky chart is highlighted):

```
// creating a row for each object in the data
var rows = d3.select("tbody").selectAll('tr')
  .data(data)

rows.enter().append("tr").attr('data-index',function(d,i){return i});
rows.on('mouseover', mouseover).on('mouseout', mouseout);
rows.exit().remove();
```

then a cell for each column in the rows is created and finally the function is closed

```
// creating a cell for each column in the rows
var cells = rows.selectAll("td").data(function(row){
      return columns.map(function(key){
            return {key:key, value:row[key]};
      });
  });
  cells.enter().append("td");
  cells.text(function(d){ return d.value; });
  cells.exit().remove();
  // closing the function:
 }) //json
} //function
```

The interaction on mouseover and clicking are described in these last simple functions: while a star or a table row is selected, the corresponding table row or circle is highlighted

```
function mouseover(d,i){
   var filter1= d3.selectAll('circle').filter(function(e,j){ return i===j });
   var filter2= d3.selectAll('tr').filter(function(e,j){ return i===j });
   filter1.attr('class','mouseover');
   filter2.attr('class','mouseover');
}
```

```
function mouseout(d,i){
   var filter1= d3.selectAll('.mouseover');
   filter1.attr('class',null);
}
```

the table is scrolled until the selected item becomes visible

```
function mousescroll(d,i){
   mouseover(d,i);
   $('tbody').scrollTop( $('tr[data-index="'+i+'"]').offset().top - $('tbody').offset().top +
   $('tbody').scrollTop() );
}
```

and finally, while passing the cursor on the color legend, all the stars in the selected range of color index are highlighted:

```
function mouserange(d,i){
   var filter1= d3.selectAll('circle').filter(function(e) { return Math.abs((e.B-e.V).toFixed(2)-d)  <step  });
   var filter2= d3.selectAll('tr').filter(function(e) {    return Math.abs((e.B-e.V).toFixed(2)-d)    < step });
   filter1.attr('class','mouseover');
   filter2.attr('class','mouseover');
}
```

A live example of this code can be seen at http://ross.iasfbo.inaf.it/~gloria/web-examples/valencia/.

4 EXAMPLES

In this section we present two websites that host tools we built for educational purposes by using the technologies described in this paper. Both sites can be displayed in 6-7 different languages and provide online tutorials. Educational reference documents are available for download on the GLORIA portal.

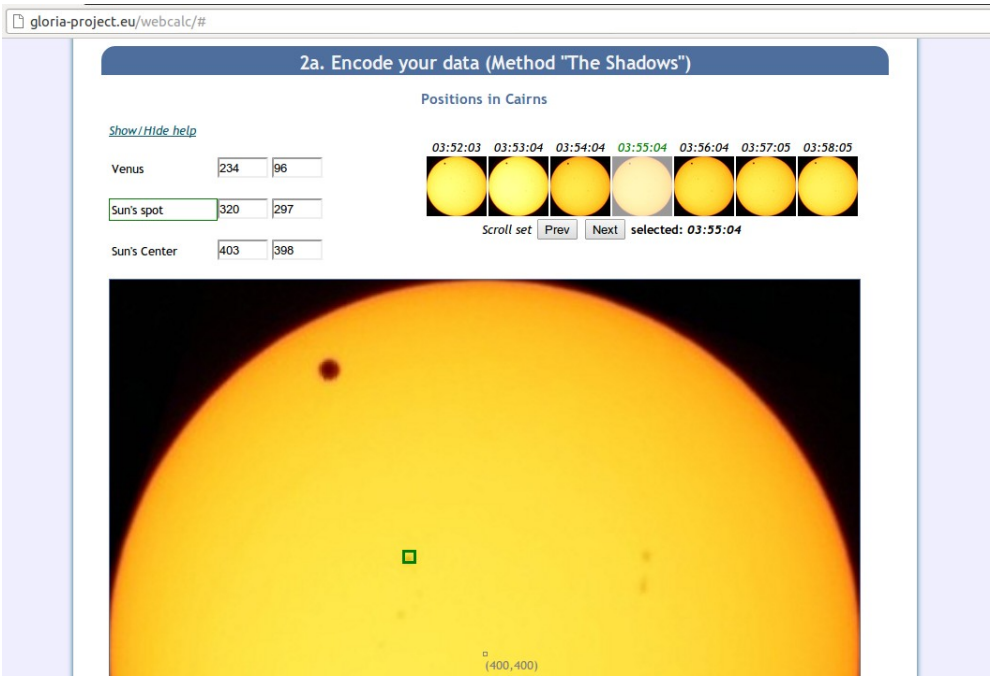


Figure 2. Venus webcalc website, available at **gloria-project.eu/webcalc/**.

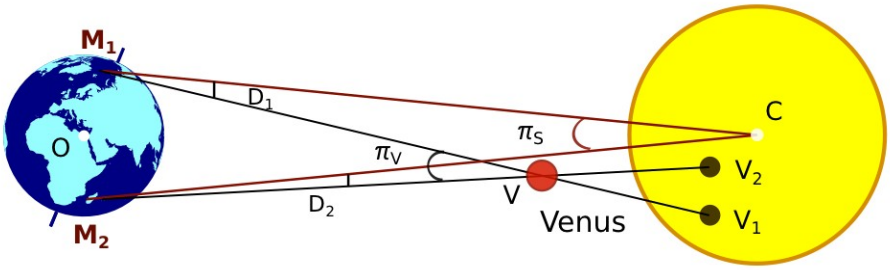


Figure 3. Schema showing the Venus transit visibility from two different sites. Angles used for the calculations of the Earth-Sun distance are shown.

4.1 Venus webcalc

This tool uses real images of the Sun taken during the last transit of Venus of June 5th, 2012 (see Fig. 2 from two distant sites on Earth. In this case from Cairns (Australia) and Sapporo (Japan). Because the two Earth site-Venus line of sights fall on different location on the Sun (see Fig. 3), trigonometric calculations can be used to calculate the Sun-Earth distance. Students can use the web tool to calculate the distance as follows: from each of the two available sets, choose an image checking that time markers are as close as possible (in principle they should be identical). For each of the two images click on the position 1. of Venus, 2. of a sunspot, and 3. of the centre of the Sun. Submitting the data, the Earth-Sun distance is calculated by triangulation (parallax). Skilled users are capable to pinpoint the three positions with an error not greater than 1 pixel. This allows to calculate the distance (of 150 million km) with an accuracy of 5-10%. All the data are managed by a relational database.

4.2 Eclipse meteo

This website makes a large use of the new web technologies described in this paper. The webtool uses real meteorological data taken in Australia during the last total solar eclipse of November 13th, 2012 (see Fig. 4). In particular it allows students to interactively calculate the time lag between observed changes in luminosity and ambient temperature. In fact an interesting effect that occurs during the course of an eclipse, more remarkable in a total eclipse, is the decrease of the environmental temperature due to the decrease of the solar radiation or ambient brightness. The interesting thing is that the minimum of the temperature drop does not occur instantaneously when the Sun is completely covered, but it occurs after a time ranging from 2 to 20 minutes. This time delay depends on many factors, such as the time of day when the eclipse occurs, the presence of nearby bodies of water such as a lake or sea, proximity to wooded areas. However the effect is easily measurable. Our webtool provides a text and a video tutorial, with subtitles. Luminosity and temperature data sets are plotted in the same plot area but the corresponding axes range can be easily (and intuitively) expanded or scrolled in a independent way to better identify the two minima. The user is then asked to guess where the minima of the luminosity and temperature curves would be if we had a continuous monitoring rather than a 1 minute sampling. Two crosshairs placed on the middle of the two minima caused by the eclipse are used to mark these points. The corresponding minimum luminosity and temperature plus the time lag between them is shown. Finally the user is asked to fill a simple form with position and personal data. On submitting the results, simple bar charts with statistics (see Fig. 4) is displayed showing how the inserted data compare with all those present in the database.

5 CONCLUSIONS

We discussed what we believe is to be considered best practice to build educational websites. We presented the latest standards and the main concepts that should be applied when designing attractive, simple and well conceived browser-based educational tools. This is particularly interesting in scientific fields like astronomy.

To give the readers an insight to these new technologies and to understand their great potential, we provided a step-by-step code tutorial implementing a basic interactive browser of the bright stars catalogue [1].

Taking advantage from these technologies, within the GLORIA project we built two websites for educational purposes and presented their main features and use.

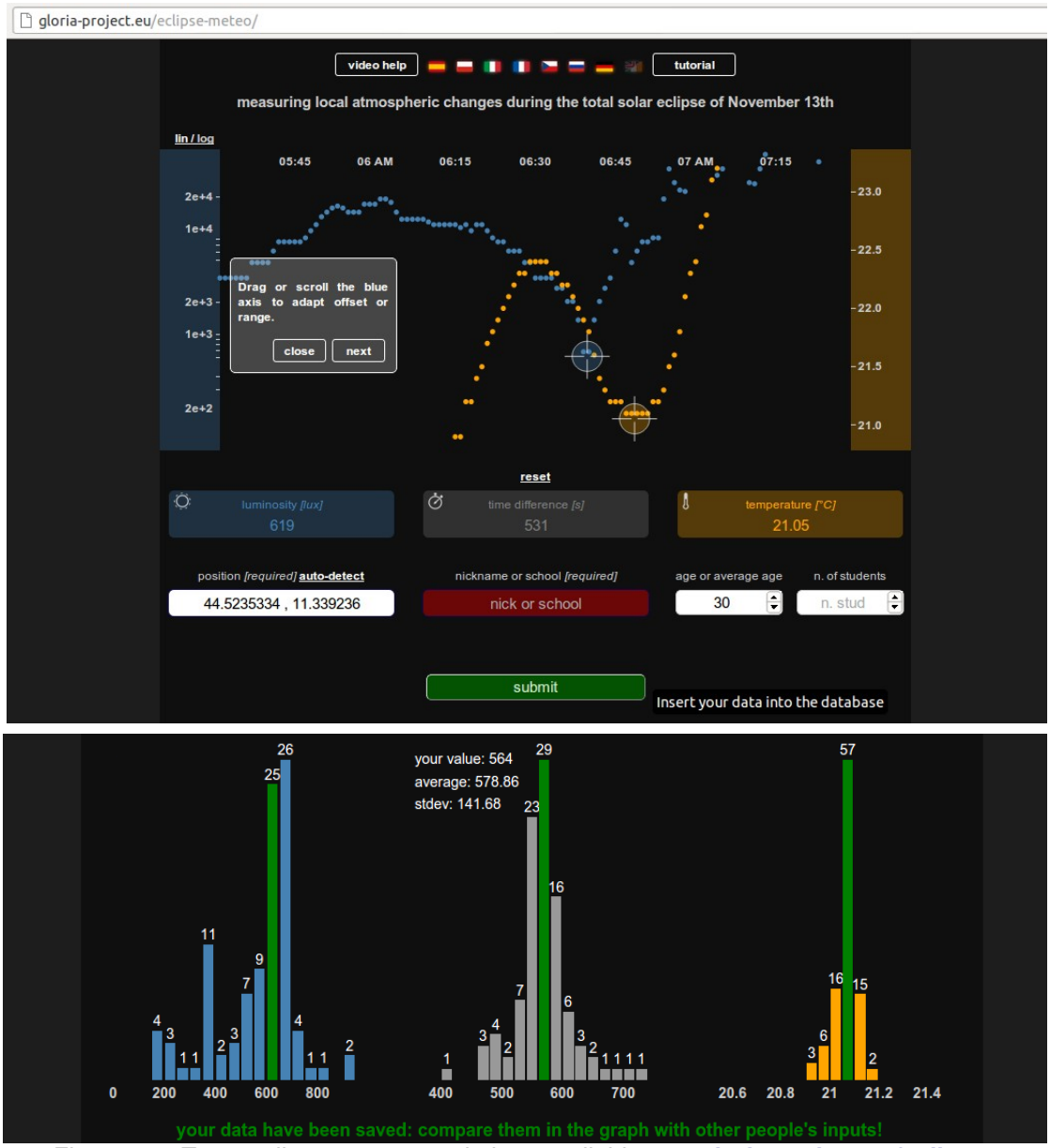


Figure 4. Top: eclipse meteo website, available at **gloria-project.eu/eclipse-meteo/**. Bottom: the statistics displayed after the form submission.

Acknowledgments: GLObal Robotic telescopes Intelligent Array for e-Science (GLORIA) is a project funded by the European Union Seventh Framework Programme (FP7/2007-2012) under grant agreement number 283783.

REFERENCES

[1] Hoffleit, D. 1964, “Catalogue of bright stars, 3rd revised”, New Haven, Conn., Yale University Observatory

[2] Bostock, M., Ogievetsky, V., Heer, J., 2011, “D3: Data-Driven Documents”, IEEE Trans. Visualization & Comp. Graphics (Proc. InfoVis)