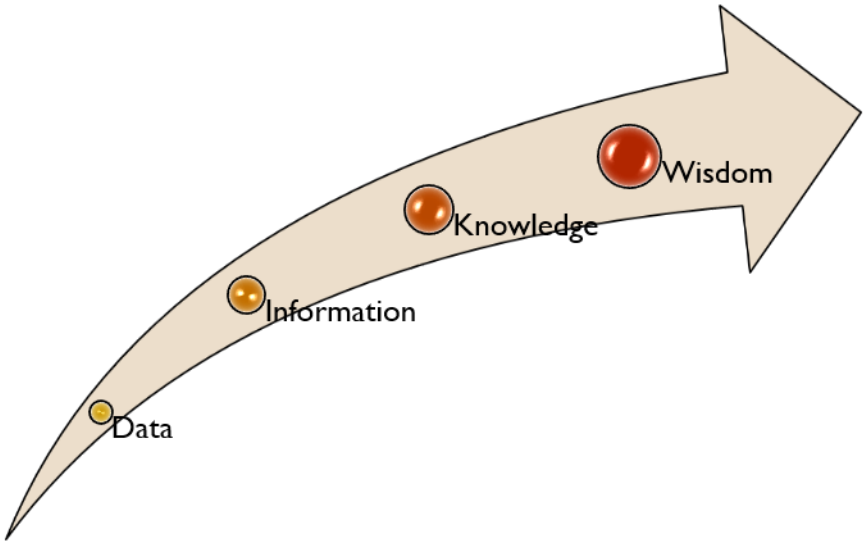

COMO VISUALIZAR LOS GRÁFICOS DE ALTAIR EN HTML

Visualización

ALTAIR + HTML

Mireia Ribera
Postgrado Data Science

Altair en HTML 2
Herramientas 2
Editor de texto 2
Un servidor 2
Como trabajar en HTML con los gráficos Altair ..... **¡Error! Marcador no definido.**

Altair en HTML

Herramientas

Para editar el código HTML necesitarás un editor de texto. Para publicar las páginas web necesitarás un servidor.

Editor de texto

Un editor de texto como Notepad++, Sublime o Atom es perfectamente apropiado para editar páginas web. Algunos tienen funcionalidades avanzadas de HTML. Elige el que más te guste. En cualquier caso, trabaja siempre con codificación UTF-8.

Una vez hayas escogido el editor, mira las extensiones y plugins de que dispone porque pueden ser de mucha utilidad. Por ejemplo, Notepad++ ofrece un plugin Javascript muy interesante (*http://www.sunjw.us/jstoolnpp/index.php*) y un plugin XML

Un servidor

Mientras no incluyas javascripts externos en tus páginas podrás probarlos directamente en el navegador, pero cuando quieras visualizar gráficos que necesiten una librería externa, seguramente necesitarás publicarlos en un servidor (un ordenador permanente contectado a Internet con un programa especial para responder a peticiones de páginas web).

La buena noticia esque puedes simular un servidor con Python. Desde el terminal puedes iniciar un servidor Pyton con:
>python –m http.server

Alternativamente también puedes descargar un modulo Node.js que ejecuta un servidor web:
>npm install-g http-server
Y tras ello, desde el terminal ejecutar:
>http-server

Recuerda que debes iniciar el servidor en el directorio que contiene tus páginas HTML o en un directorio superior que lo contenga, para poder navegar hasta él.

Una vez ejecutado el servidor (con Python o con Node.js) podrás ver tus páginas en el navegador en la dirección: http://localhost:8080/ or http://localhost:8888/ o en la URL que te indique el terminal cuando inicias el servidor.

Be aware that the server initiates on the folder from which you executed it, so you must navigate to your html files from there, and you will not be able to reach hierarchically superior folders.

El código HTML

Desde Altair puedes guardar gráficos en HTML. Las páginas generadas se pueden visualizar directamente. En el código se pueden reconocer los elementos habituales de una página web como html, head, body, style, script

En la cabecera se encuentran referencias a las librerías javascript externas necesarias para Altair:

```
  <script type="text/javascript"
src="https://cdn.jsdelivr.net/npm//vega@5"></script>
```

```
  <script type="text/javascript" src="https://cdn.jsdelivr.net/npm//vega-
lite@3.4.0"></script>

  <script type="text/javascript" src="https://cdn.jsdelivr.net/npm//vega-
embed@4"></script>
```

También estilos para los botones y los mensajes de error:

```
  <style>
    .vega-actions a {
        margin-right: 12px;
        color: #757575;
        font-weight: normal;
        font-size: 13px;
    }
    .error {
        color: red;
    }
  </style>
```

Y en el body, como único elemento HTML, un div sin nada dentro:

```
<div id="vis"></div>
```

Seguido de un script muy largo, en el que:

- Se da la especificación del gráfico:

```
var spec = {"config": {"view": {"width": 400, "height": 300}, "mark": {"tooltip":
null}}, "layer": [{"data": {"url": "bcn.json", "format": {"feature": "bcn",
"type": "topojson"}}, "mark": {"type": "geoshape", "fill": "lightgray", "stroke":
"white"}, "height": 300, "projection": {"type": "mercator"}, "width": 500},
{"data": {"url": "accidents_2017.csv"}, "mark": {"type": "circle", "fill":
"#ff0000", "opacity": 0.5}, "encoding": {"latitude": {"field": "Latitude",
"type": "quantitative"}, "longitude": {"field": "Longitude", "type":
"quantitative"}, "size": {"value": 3}, "tooltip": {"type": "nominal", "field":
"Serious injuries"}}}], "$schema": "https://vega.github.io/schema/vega-
lite/v3.4.0.json"};
```

- Se definen las opciones vega-lite

```
var embedOpt = {"mode": "vega-lite"};
```

- Y hay una función para mostrar los errores si se producen

```
function showError(el, error){
          el.innerHTML = ('<div class="error" style="color:red;">'
                          + '<p>JavaScript Error: ' + error.message + '</p>'
                          + "<p>This usually means there's a typo in your chart
specification. "
                          + "See the javascript console for the full
traceback.</p>"
                          + '</div>');
          throw error;
      }
```

Finalmente, la variable *el* coge el elemento DOM llamado vis

```
      const el = document.getElementById('vis');
```

y le inserta la especificación del gráfico con las opciones

```
vegaEmbed("#vis", spec, embedOpt)
```

Estos elementos se llaman igual para todos los gráficos.

Si lo que que queremos es combinar dos o más gráficos en una misma página, deberemos mezclar los códigos y poner identificadores únicos (vis2, spec2, el2; vis3, spec3, el3…).

```
<div id="vis"></div>
<div id="vis2"></div>
<script>
var spec = {"…
var spec2 = {"…
var embedOpt = {"mode": "vega-lite"};// same code for both charts; otherwise
change
function showError(el, error){…    //again the same for both charts
const el = document.getElementById('vis');
const el2 = document.getElementById('vis2');
vegaEmbed("#vis", spec, embedOpt)
    .catch(error => showError(el, error));
vegaEmbed("#vis2", spec2, embedOpt)
    .catch(error => showError(el2, error));
```

.