

## Homework 2

This homework is due on *Monday, February 8th 2016 at 11:59 pm.*

In this assignment, you'll use HTML, CSS and JavaScript to create a small dashboard for a local pizza delivery company. It should help the company to get meaningful insights into their collected data and to evaluate their performance. We provide a small framework, based on *Bootstrap*, and two example datasets.

In the design part of this homework you have to critique a visualization (you can choose between two given examples) and you have to create a few sketches, similar to our previous lecture.

This homework assumes that you have read and programmed along with chapter 3 (up to page 49) in *D3 - Interactive Data Visualization for the Web* by Scott Murray.



# 1) Complete lab 2

---

Please put all completed files from Lab 2 in the `lab` directory on Vocareum and submit it with the rest of this homework.

## 2) PizzaWorld - Insights & Statistics

---

### Framework

We are providing a small template (based on *Bootstrap*) for this homework. It is quite similar to the framework which we have used in our second lab and should help you to get started. You can download it [here](#).

You will need to extend the following files:

- `index.html`
- `css/style.css`
- `js/main.js`

### Data

We have included two datasets from *PizzaWorld* in the framework. One table has 1198 rows with pizza deliveries and the other table has 200 rows with corresponding customer feedback.

The data (arrays with JSON objects) are already stored in these two global variables:

- `deliveryData`
- `feedbackData`

The variables are accessible in the JS file `main.js`

To get started, write the variables to the web console and inspect them.

### Delivery Data

- `delivery_id`
- `date`
- `area` (Boston | Cambridge | Somerville)
- `price` (in USD)
- `count` (number of delivered pizzas)
- `driver`

- `delivery_time` (minutes)
- `temperature`
- `drinks_ordered` (true | false)
- `order_type` (web | phone)

## Feedback Data

- `delivery_id` (matches the deliveries)
- `punctuality` (medium | low | high)
- `quality` (medium | low | high)
- `wrong_pizza` (true | false)

## Implementation

### 1. Download framework

If you haven't done so already, please download the [framework for homework 2](#).

### 2. Display dataset summary

Extract the following numbers from the provided data and append them with meaningful labels to the HTML document (part 2 in function `createVisualization()`):

- Number of pizza deliveries
- Number of all delivered pizzas (*count*)
- Average delivery time
- Total sales in USD
- Number of all feedback entries
- Number of feedback entries per quality category: *low*, *medium*, *high*

→ You should update the DOM dynamically with JavaScript!

*Feel free to use plain JavaScript methods or the library jQuery to update the content. The JS framework jQuery is already included because it is required for Bootstrap. You can learn more about jQuery on these websites: [jquery.com](http://jquery.com) or [w3schools.com/jquery/](http://w3schools.com/jquery/).*

Make sure you test your code after you have implemented this part! Once your code is working, consider your code structure and code design:

- a) If your code is already longer than a couple of lines, extract it into a separate function and call that function within `createVisualization()`.

- b) Think about calculating the above numbers at the same time (e.g., inside a single loop), if possible. That will make the code shorter and easier to read, as well as faster.
- c) Make sure you use consistent spacing, layout, self-explanatory variable names, and code comments! If you do this as you progress in your coding project, this will save you a lot of time later!

### 3. Call the function: *renderBarChart(deliveryData)*

If you call *renderBarChart()* our script will automatically group (and count) the deliveries per day. Subsequently a bar chart will be added to the container: `#chart-area`.

Requirements:

- You need a div-container with the id `#chart-area` in your HTML document
- As a **function parameter**, you have to include a JSON array with deliveries (each delivery is an object and must include at least the property: `date`). Try to call *renderBarChart()* (inside `createVisualization`) with the variable `deliveryData`.

→ You should now see a basic bar chart in your webpage.

### 4. Filter the data before drawing the bar chart

In the next task, you should filter the `deliveryData` array depending on the selected options, before calling *renderBarChart()*.

a. Add two **select-boxes** to your HTML document. The user should be able to select:

- `area` (*All, Boston, Cambridge or Somerville*)
- `order type` (*All, Web or Phone*).

Make sure to set the callback function of the select boxes to the function that updates and creates our visualization!

b. If the user selects an option, filter the variable `deliveryData` accordingly and then call *renderBarChart()* again. Every time you call this function, it will analyze all changes in your array and update the chart with D3.

It is good practice to do a sanity check on the result you get after filtering or processing data. Take a look at the original data, do the numbers make sense? Are there missing values? This is especially important when dealing with real-world data that has not been cleaned and prepared for easy use in a homework ;-)

### 5. Global data filtering

In the previous step you have implemented a function which reacts to the user input (change of a select-box value) and updates the chart immediately.

Additionally, these filter options should also affect the displayed dataset summary that you implemented earlier.

→ If the user filters the data by a specific `area` or `order type` you should **update the bar chart and the display of the summary data** from step (2).

*Hint: The data are stored in two separate variables. Make sure your filtering algorithm influences customer feedback as well as order data.*

## 6. Style your webpage using CSS.

We already provided a simple css file in our template. Adjust either the CSS or the html elements you dynamically generate to create a visually pleasing and easy to read webpage.

# 3) Design Analysis / Critique

---

- Choose **ONE** of the visualizations (*A* or *B*)
- Evaluate the visualization based on lie factor, data-ink ratio and graphical integrity
- Answer the following questions:
  1. Who is the audience?
  2. What questions does this visualization answer?
  3. What design principles best describe why it is good / bad?
  4. Why do you like / dislike this visualization?
  5. How could the data-ink ratio be improved? And can you suggest any other improvements?

*Please do not use more than 400 words (cumulative) to evaluate the visualization and answer all questions. You will have to submit your design critique as a PDF document.*

# Visualization A

<http://www.nytimes.com/interactive/2008/05/03/business/20080403SPENDINGGRAPHIC.html>

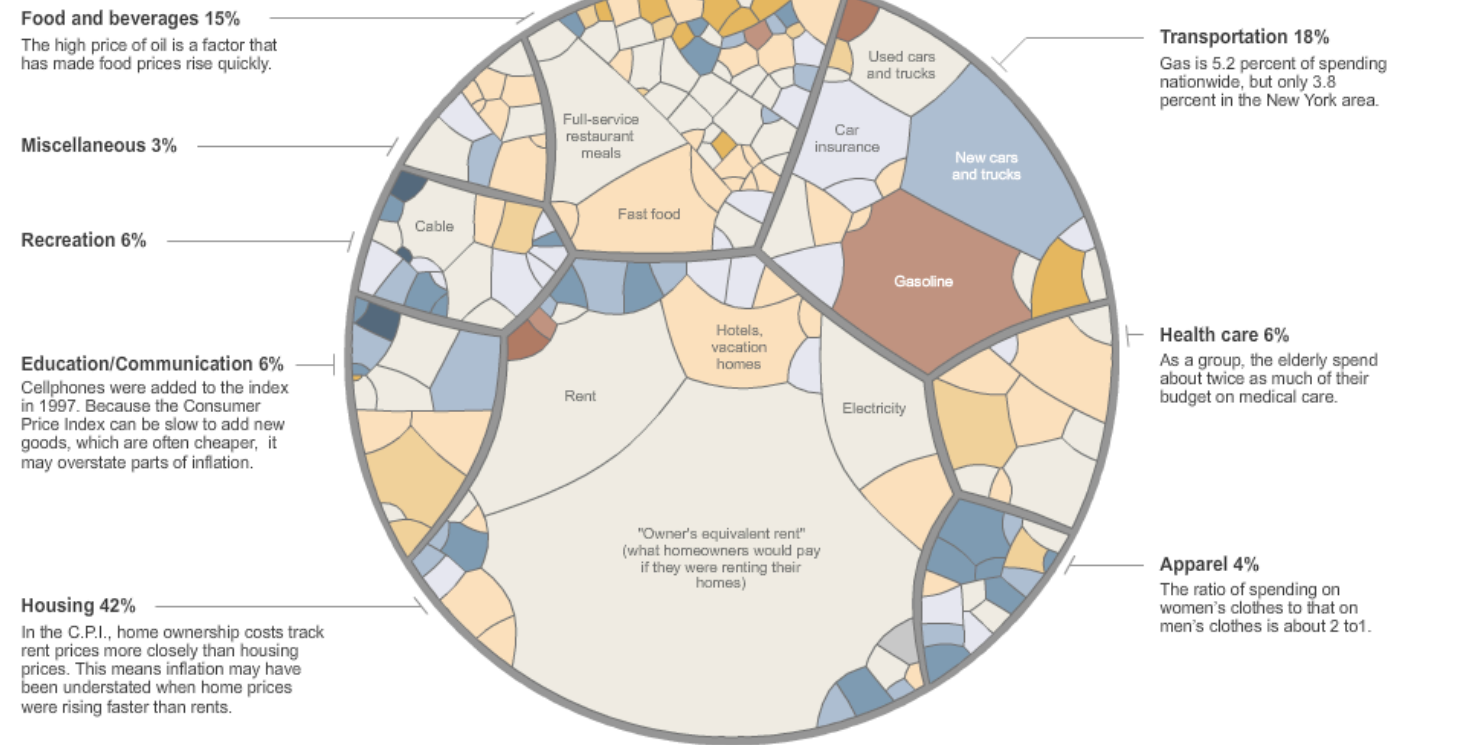
## An Average Consumer's Spending

Each shape below represents how much the average American spends in different categories. Larger shapes make up a larger part of spending.

Color shows change in prices from March 2007 to March 2008



ZOOM IN ZOOM OUT



Sources: Bureau of Labor Statistics; Michael Balzer, University of Konstanz (Germany)

Matthew Bloch, Shan Carter and Amanda Cox/The New York Times

# Visualization B

<http://visual.ly/nobels-no-degrees-english>

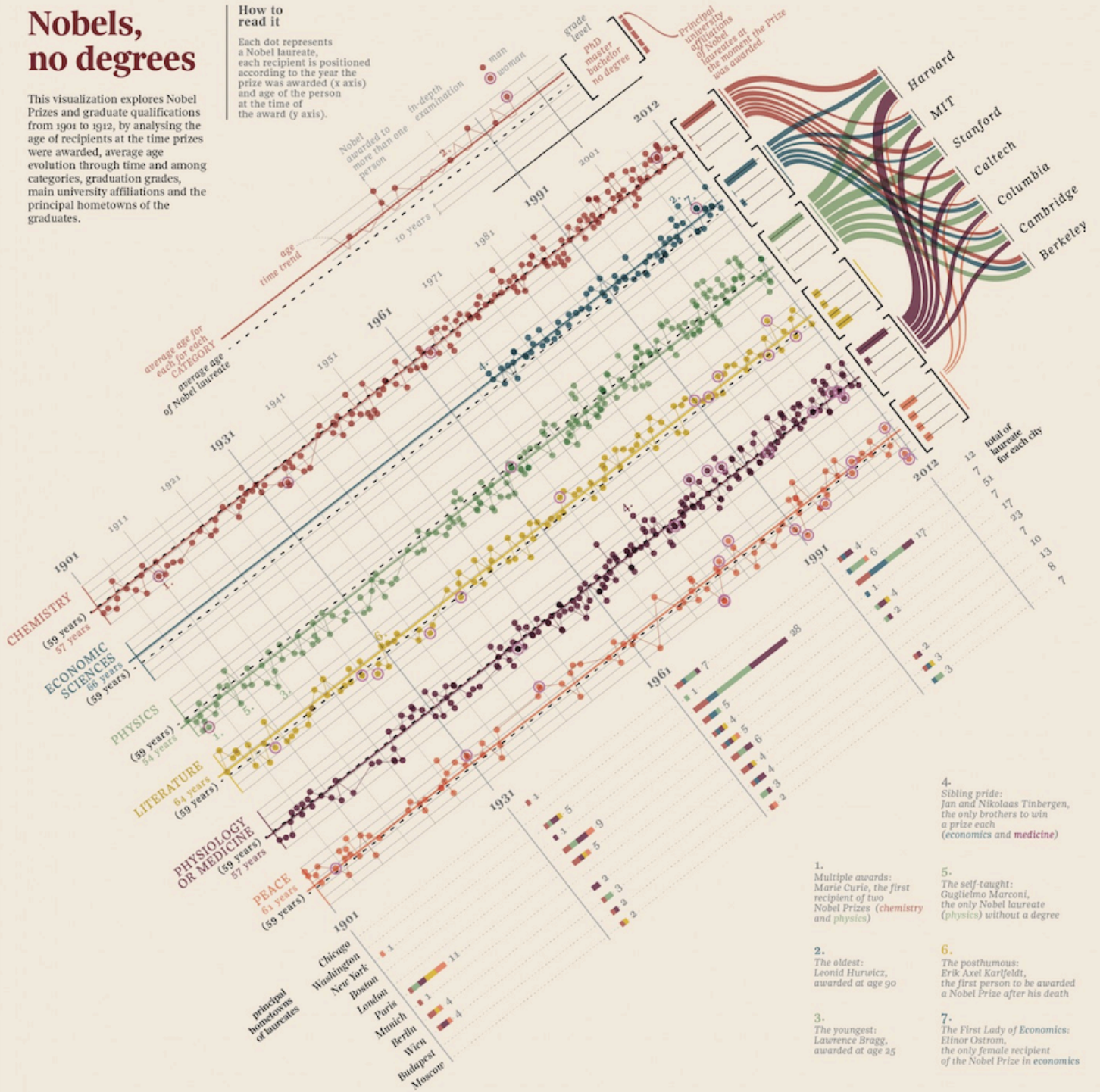


# Nobels, no degrees

This visualization explores Nobel Prizes and graduate qualifications from 1901 to 1912, by analysing the age of recipients at the time prizes were awarded, average age evolution through time and among categories, graduation grades, main university affiliations and the principal hometowns of the graduates.

## How to read it

Each dot represents a Nobel laureate, each recipient is positioned according to the year the prize was awarded (x axis) and age of the person at the time of the award (y axis).



## 4) Design Creation

Create 5 sketches to visualize the following data and name the visual dimensions you are using to encode the information:

<b>A:</b>	12	23
<b>B:</b>	32	17

*Scan your sketches and create a PDF with all results.*

## 5) Submit Homework on Vocareum

---

1. **Upload your JS homework under 'work/hw/implementation'.** (Depending on your project structure you might have to create new directories in Vocareum. You can upload multiple files at once into the same directory. We do not encourage a flat data hierarchy, but due to the limitations of Vocareum we will also not deduct points for a flat file structure.)
2. **Upload 2 PDF files (design critique and design creation) under 'work/hw/design'**
3. **Upload the completed lab under 'work/lab'**
4. **For DCE students, upload your lecture activities under 'work/lecture'**

For your final submission you will have to:

- Click the 'submit' button Double-check your 'latest submission', check that all the visualizations are working as you expect. You can run them directly in Vocareum and look at them in a new window
- Upload a teaser image of your homework (under 'Action'/'upload gallery thumbnail'. This teaser will show up in the classes gallery and should motivate other students to look at your solution in more detail.

After the homework deadline you will be able to:

- Look at the gallery with all other student submissions
- Give peer feedback or comments to other students (Please use proper manners and constructive criticism!)
- Give stars to other students' work (this will not influence grading!)