

# Interactive charts

*Combine HighCharts with PX files*

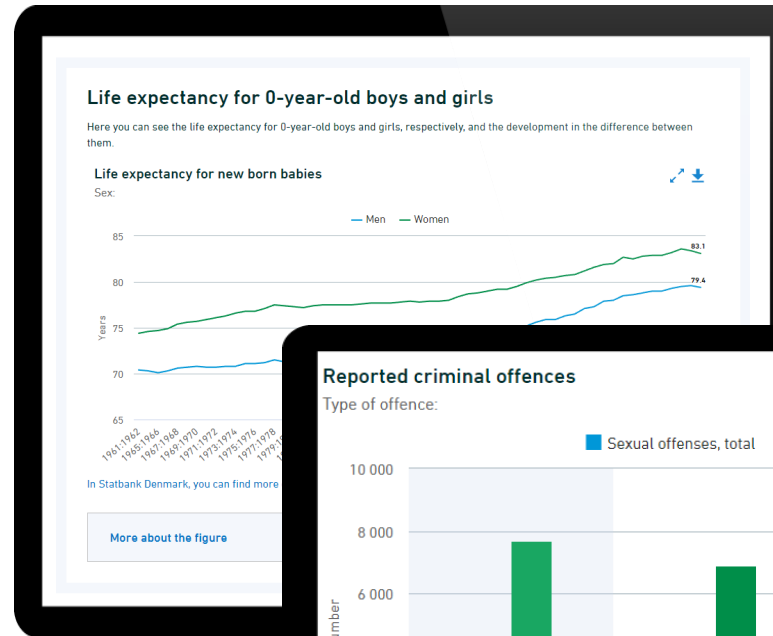
Stefan Jul Gunnensen, Chief Adviser  
Statistics Denmark  
PX Meeting, September 2023



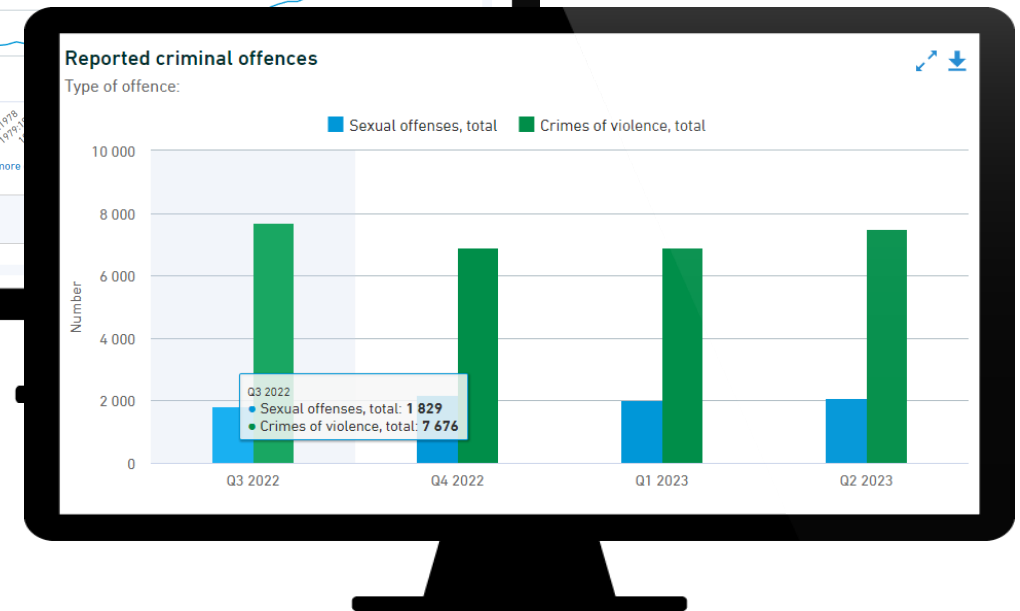


# More focus on interactive charts from 2019 →

- Needed a general solution for interactive charts on Statistics Denmark's website, [www.dst.dk](https://www.dst.dk)
- Lars Knudsen developed a prototype (again)
- PX files as data source: Self contained format with data and metadata
- Based on HighCharts (JavaScript library) combined with Statistics Denmark's StatBank API
  - <https://www.highcharts.com>
  - <https://api.statbank.dk>



<https://www.dst.dk/en/Statistik/emner/borgere/befolkning/middelevetid>



<https://www.dst.dk/en/Statistik/emner/sociale-forhold/kriminalitet>

# Custom library dstChart.js published in late 2019

- A lot of settings in HighCharts
- A lot of common settings for each chart depending on type
- Some custom settings needed for each chart, e.g. title/subtitle
- **dstChart.js** was a good solution
- Delivered predefined design with multiple options for data sources and customizing individual charts
- 1<sup>st</sup> version was in production ultimo 2019 on [www.dst.dk](http://www.dst.dk) and [www.statbank.dk](http://www.statbank.dk)

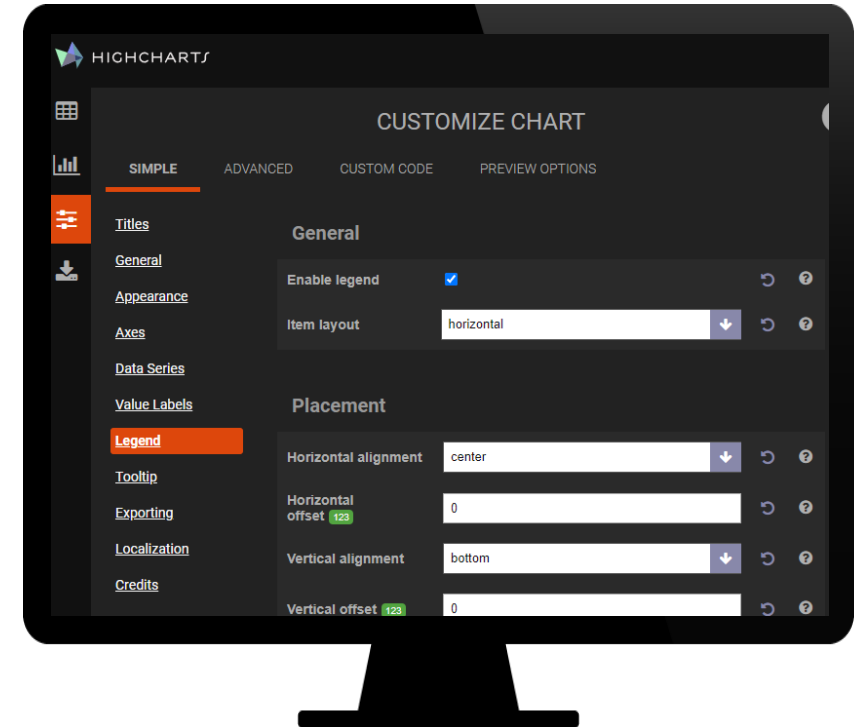


```

HIGHCHARTS.
Highcharts v11.1.0 Namespaces Classes Interfaces
global: {...}
lang: {...}
Highcharts.chart({
  accessibility: {
    announceNewData: {...}
    customComponents: undefined
    description: undefined
    enabled: true
    highContrastTheme: undefined
    keyboardNavigation: {...}
    landmarkVerbosity: "all"
    linkedDescription: "[data-highcharts-chart="
    point: {...}
    screenReaderSection: {...}
    series: {...}
    typeDescription: undefined
  }
  annotations: [{
    animation: {...}
    controlPointOptions: {...}
    crop: true
    draggable: xy
    events: {...}
    id: undefined
    labelOptions: {...}
    labels: [...]
    shapeOptions: {...}
    shapes: [...]
    visible: true
    zIndex: 6
  }

```

HighChart's API reference:  
<https://api.highcharts.com/highcharts/>

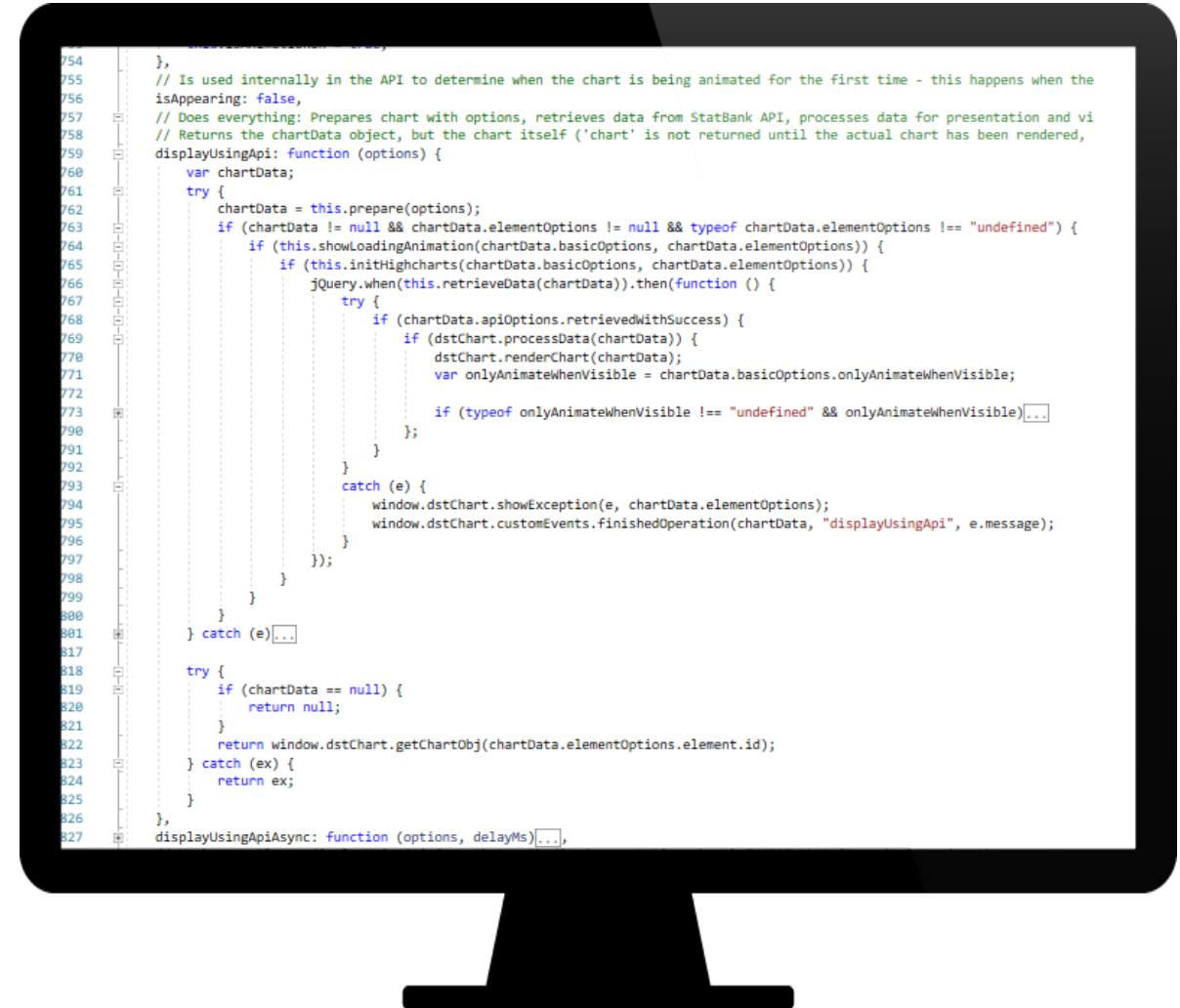


HighChart's own editor with a small selection of options:  
<https://editor.highcharts.com/full.html>

# Lots of details in the work with dstChart.js

- Development time spent on HighCharts API, data processing, use of styles, specific design issues etc.
- Currently approx. 4,000 lines of code
- A minimal simple visualization requires only this:
  - references to a **dstChart.js bundle**
  - **id** for a container in HTML (e.g. a div)
  - A single call in JavaScript for the HTML

```
dstChart.displayUsingApi({  
  elementOptions: { elementId: "chart" },  
  apiOptions: { pxsId: 233172, method: "saved" }  
});
```

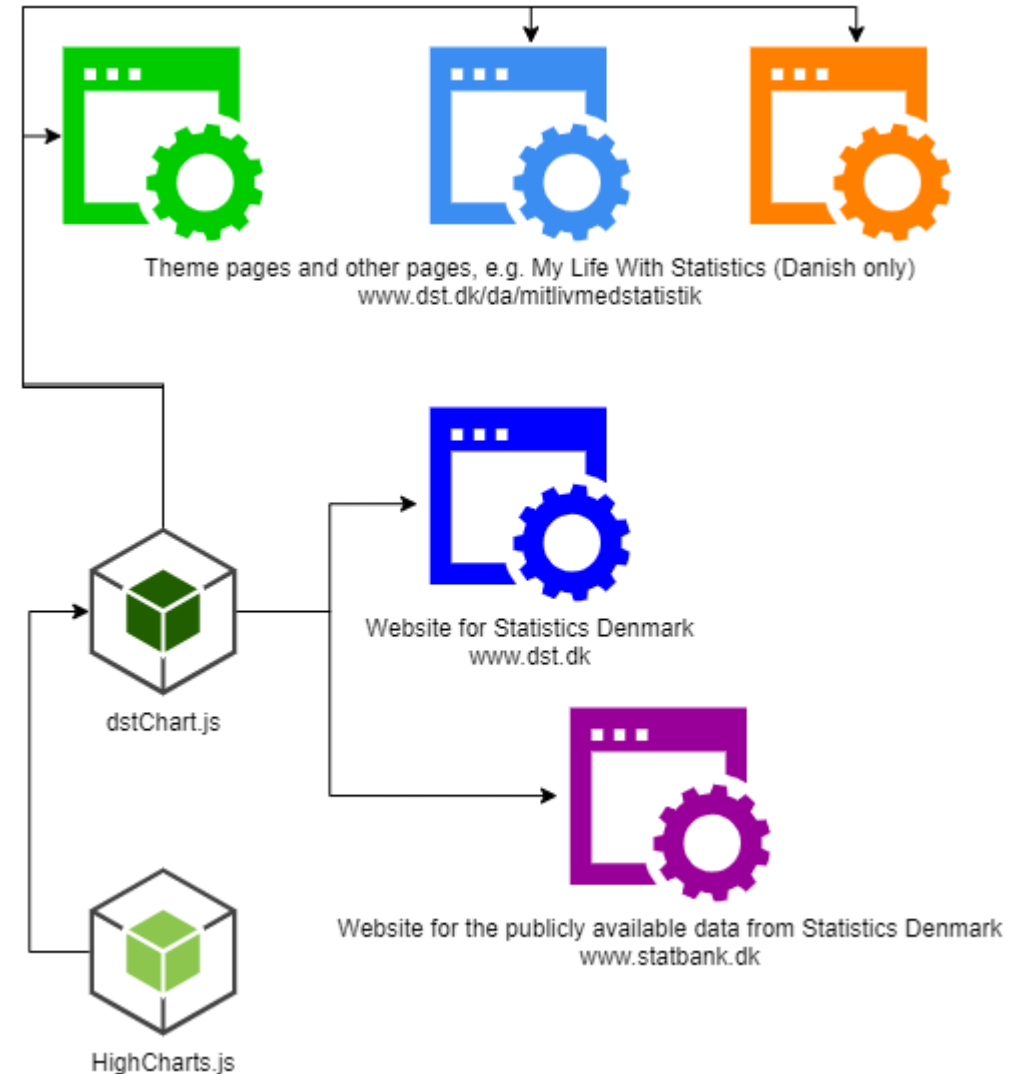


```
754     },  
755     // Is used internally in the API to determine when the chart is being animated for the first time - this happens when the  
756     isAppearing: false,  
757     // Does everything: Prepares chart with options, retrieves data from StatBank API, processes data for presentation and vi  
758     // Returns the chartData object, but the chart itself ('chart' is not returned until the actual chart has been rendered,  
759     displayUsingApi: function (options) {  
760         var chartData;  
761         try {  
762             chartData = this.prepare(options);  
763             if (chartData != null && chartData.elementOptions != null && typeof chartData.elementOptions != "undefined") {  
764                 if (this.showLoadingAnimation(chartData.basicOptions, chartData.elementOptions)) {  
765                     if (this.initHighcharts(chartData.basicOptions, chartData.elementOptions)) {  
766                         jQuery.when(this.retrieveData(chartData)).then(function () {  
767                             try {  
768                                 if (chartData.apiOptions.retrievedWithSuccess) {  
769                                     if (dstChart.processData(chartData)) {  
770                                         dstChart.renderChart(chartData);  
771                                         var onlyAnimateWhenVisible = chartData.basicOptions.onlyAnimateWhenVisible;  
772                                         if (typeof onlyAnimateWhenVisible != "undefined" && onlyAnimateWhenVisible)[]  
773                                         };  
774                                     }  
775                                 }  
776                             } catch (e) {  
777                                 window.dstChart.showException(e, chartData.elementOptions);  
778                                 window.dstChart.customEvents.finishedOperation(chartData, "displayUsingApi", e.message);  
779                             }  
780                         });  
781                     }  
782                 }  
783             } catch (e)[]  
784         }  
785         try {  
786             if (chartData == null) {  
787                 return null;  
788             }  
789             return window.dstChart.getChartObj(chartData.elementOptions.element.id);  
790         } catch (ex) {  
791             return ex;  
792         }  
793     },  
794     displayUsingApiAsync: function (options, delayMs)[]  
795 }
```

A very small sample of the code from dstChart.js

# Integrated in various ways across websites

- dstChart.js is useful in many scenarios
- Integrated with main website ([www.dst.dk](http://www.dst.dk)) using generated HTML from asp.net
- Integrated in our web backend in an asp.net UI editor with preview of charts
- Integrated with StatBank ([www.statbank.dk](http://www.statbank.dk)) with extra settings using classic asp
- Integrated on a standalone theme site for children using HTML and JavaScript
  - “My life with statistics”  
<https://www.dst.dk/da/mitlivmedstatistik/min-tid---mit-liv-med-statistik#chapter5>



# Processing steps of dstChart.js for each chart

- Multiple steps when dealing with a simple call.
1. **Prepare** (select element, initialize basic stuff)
  2. **Retrieve** (get data in one way or another)
  3. **Process** (Adapt PX data in various ways)
  4. **Render** (display the chart)
    - Animated when scrolled into view
    - Example: <https://www.dst.dk/en/Statistik/emner/sociale-forhold/kriminalitet>
  5. **Post processing** (custom manipulation)
- Performance is good and scales reasonable well
  - Beware of data consumption, especially on page refresh



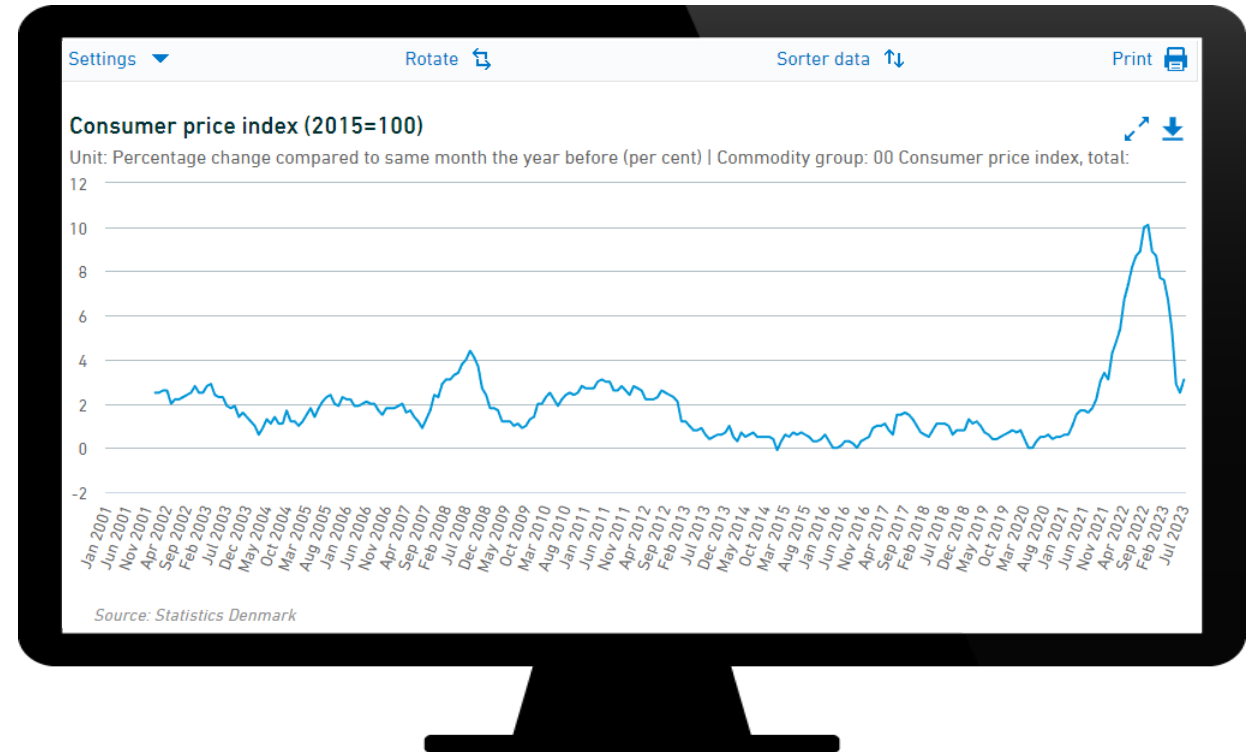


# Flexible data sources in dstChart.js

- The StatBank API is one source, but potentially not the only one
- A PX file is all that is needed

```
dstChart.displayUsingApi({
  elementOptions: { elementId: "chart" },
  apiOptions: {
    pxsPath: "http://example.com/myfile.px",
    method: "savedFile" }
});
```

- If you have a PX file, you can visualize it
- Other formats could be supported, e.g. JsonStat
  - Under consideration for the theme website “My life with statistics”



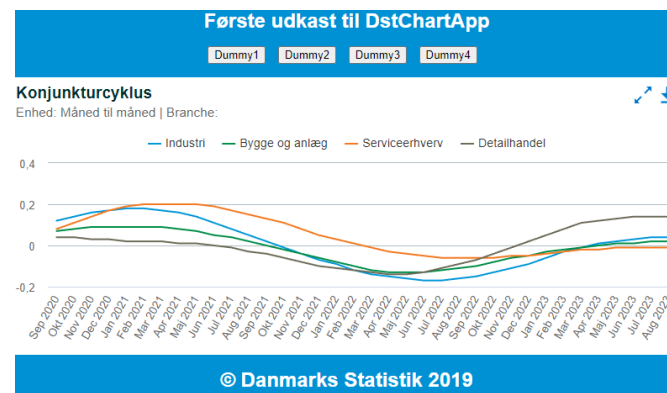
An example from Statistics Denmark's StatBank:

<https://www.statbank.dk/statbank5a/SelectVarVal/define.asp?MainTable=PRIS111&PLanguage=1&PXSid=238917&ST=ST>



# Open source and integration with other products

- Built with open source *in mind*, but not publicly available for now
- Sparse in-house documentation
- Source code could be shared, but is currently **not** on GitHub
- Some support for extending source code, but needs more work
- Easy integration of dstChart.js when using PX files
- A standalone product that could be included in other products
- Licensing for HighCharts might present a problem



Old prototype for standalone app using dstChart.js

# Possible integration of dstChart.js with PxWeb

- Currently static images, but that might change in next version of PxWeb ...?
- PxModel* seems to be used in PxWeb source code, not *PX files*
- PxApi could be used to supply dstChart.js with PX file ...
- ... or dstChart.js could add support for PxApi and saved queries

✓ About table

✓ Show result as...

✓ Edit and Calculate

✓ Save result as...

✓ Save your query

✓ Chart settings

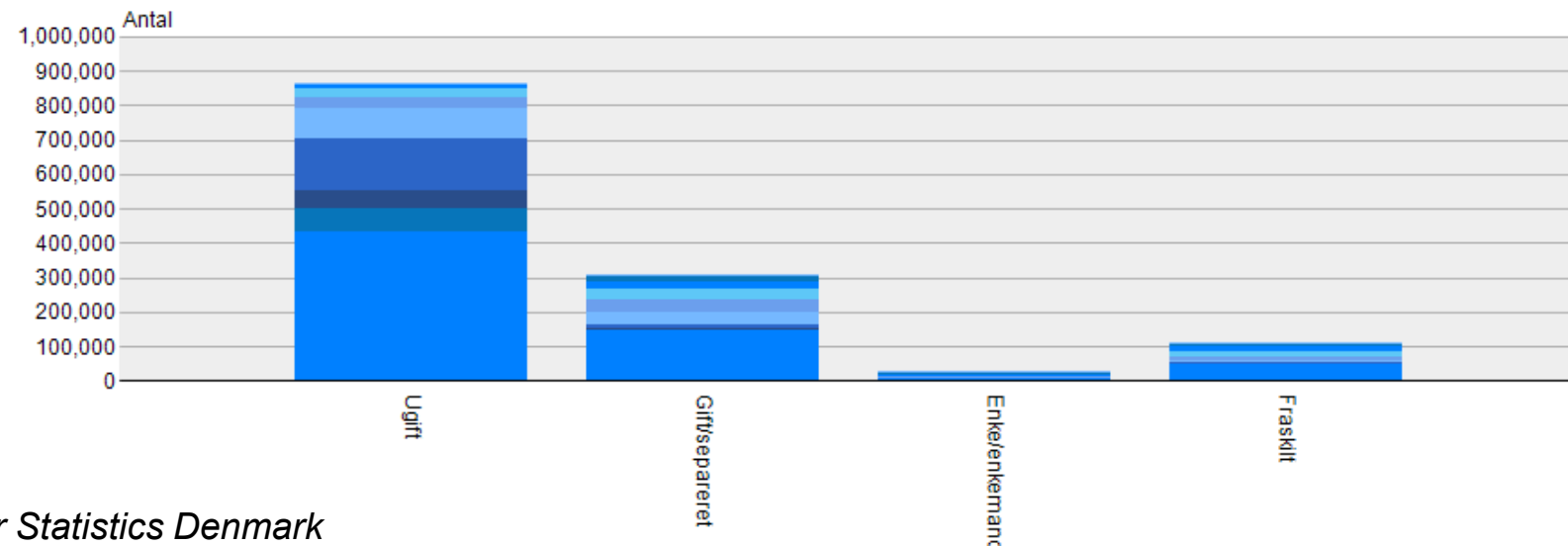
↻ Pivot manual

↻ Pivot clockwise

↻ Pivot counterclockwise

↗ Fullscreen

Folketal den 1. i kvartalet by alder and civilstand. København, 2023K3.



Source: Static chart in PxWeb for Statistics Denmark

# The future of dstChart.js

- Slowly adding new features/options and fixing small issues
- “*Need to have*” basis for development
- “*Nice to have*” would be more fun!
- Implementation of interactive charts in publications
  - Automatic generation of HTML with HighCharts using R
- Dependent on HighCharts and its features
  - HighCharts component needs regular upgrades (version 9 is used, but 11 is available)
- Multiple chart support are complex and a bit clumsy
- Performance *could* be improved
- ... but (too) few resources ...



'THE DOCTOR WILL SEE YOU MR JONES - IF YOU COULD STILL BE ILL A WEEK ON WEDNESDAY.'

# Questions

- **How do I get my hands on dstChart.js?**
  - Just download it!
    - <https://www.dst.dk/Site/Global/script/dstchart-accessibility-1.0.0-hc-9.0.1.min.js>
  - Or send me an email ([sjg@dst.dk](mailto:sjg@dst.dk))
    - If specific bundles are needed (with or without jQuery)
    - If you need example code or instructions
    - If you need the (rough) documentation
    - If you need the source code for extending or changing behavior
- **Would you kindly make it available on GitHub?**
  - It can be done with a little work.
  - It just needs to be prioritized 😊
- **Other questions?**

# Thank you for your attention!

Stefan Jul Gunnensen  
Statistics Denmark

