

# Information Visualization

## W06: Creating Data Plot - Scatter Plot

Graduate School of System Informatics

Department of Computational Science

**Naohisa Sakamoto   Akira Kageyama**

April 27, 2022

# Schedule

- W01 4/12 Guidance
- W02 4/13 JavaScript Programming
- W03 4/19 Data and Tasks
- W04 4/20 Reading Data
- W05 4/26 Marks and Channels
- W06 4/27 Creating Data Plot - Scatter plot
- W07 5/10 Visualization Idioms
- W08 5/11 Creating Data Plot - Bar/Pie/Line/Area chars

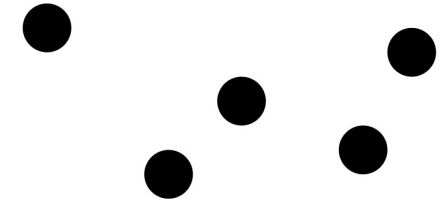
# Scatter Plot

- **Drawing Points**

- Data: W04/data.csv
- D3: d3.v6.min.js (Version 6)
- Separate JS file
  - index.html
  - main.js

```
x,y,r
20,20,10
100,50,10
70,80,10
170,30,10
150,70,10
```

CSV



HTML



JS

```
...
<script src="https://d3js.org/d3.v6.min.js"></script>
<script src="main.js"></script>
...
```

```
...
d3.csv("https://xxx.github.io/InfoVis2022/W04/data.csv")
...
```

# Example 01

- **Drawing Points**

- index.html
  - Use D3 v6
  - Specify main.js

```
<html>
  <head>
    <title>W06: Example 01</title>
  </head>
  <body>
    <script src="https://d3js.org/d3.v6.min.js"></script>
    <script src="w06_ex01_main.js"></script>
  </body>
</html>
```

w06\_ex01\_index.html

# Example 01

- **Drawing Points**

- main.js
  - Load the data
  - Call ShowScatterPlot function

```
d3.csv("https://xxx.github.io/InfoVis2022/W04/data.csv")
  .then( data => {
    ShowScatterPlot(data);
  })
  .catch( error => {
    console.log( error );
  });
...
```

w06\_ex01\_main.js

# Example 01

- **Drawing Points**

- main.js
  - Load the data
  - Call ShowScatterPlot function

```
...  
function ShowScatterPlot(data) {  
  var svg = d3.select("body").append("svg");  
  svg.selectAll("circle")  
    .data(data)  
    .enter()  
    .append("circle")  
    .attr("cx", d => d.x)  
    .attr("cy", d => d.y)  
    .attr("r", d => d.r);  
};
```

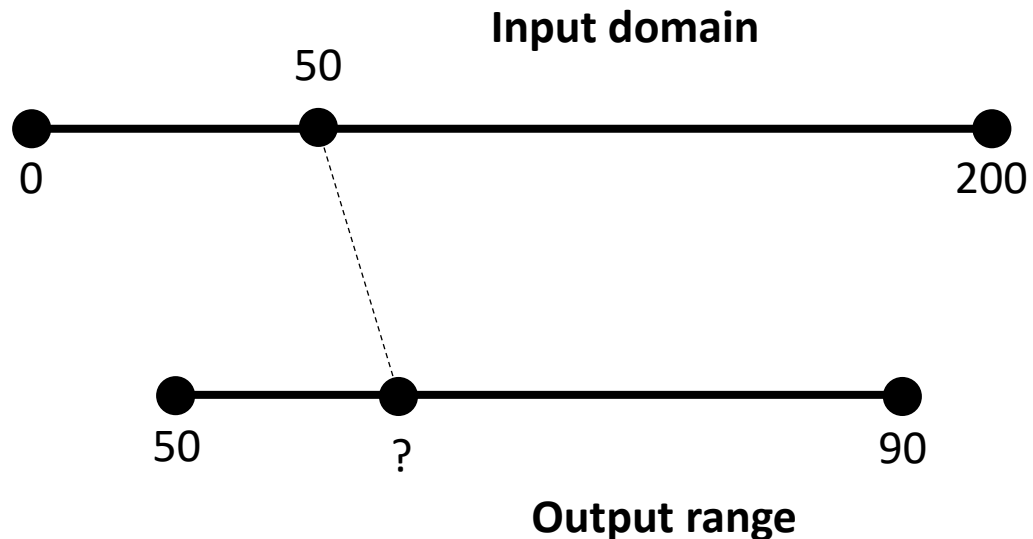
w06\_ex01\_main.js

# Scales

- **Linear scaling**

- d3.scaleLinear

- Transform an input interval (**domain**) into a new interval (**range**)



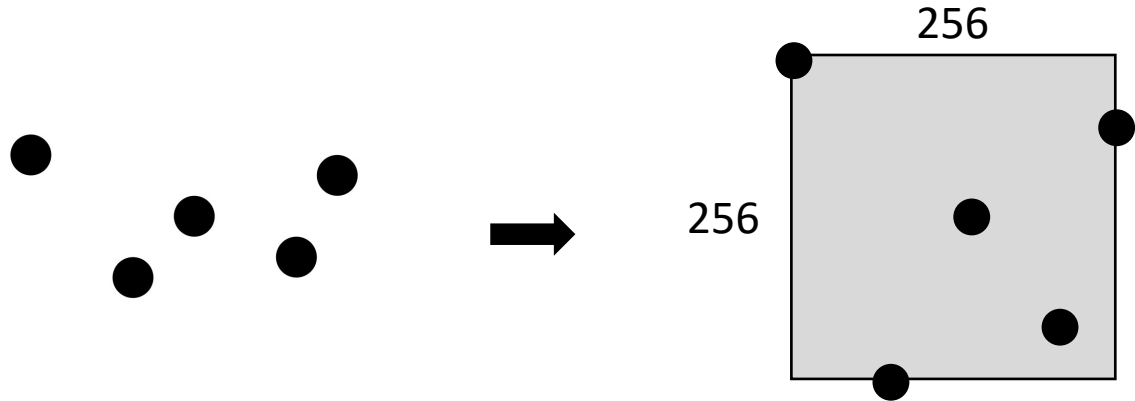
```
const linear_scale = d3.scaleLinear()  
  .domain( [0, 200] )  
  .range( [50, 90] );
```

```
const input = 50;  
const output = linear_scale( 50 );
```

# Example 02

- **Linear Scaling**

- Scaling x and y axes



```
d3.csv("https://xxx.github.io/InfoVis2022/W04/data.csv")
  .then( data => {
    // Convert strings to numbers
    data.forEach( d => { d.x = +d.x; d.y = +d.y; });
    ShowScatterPlot(data);
  })
  .catch( error => {
    console.log( error );
  });
...

```

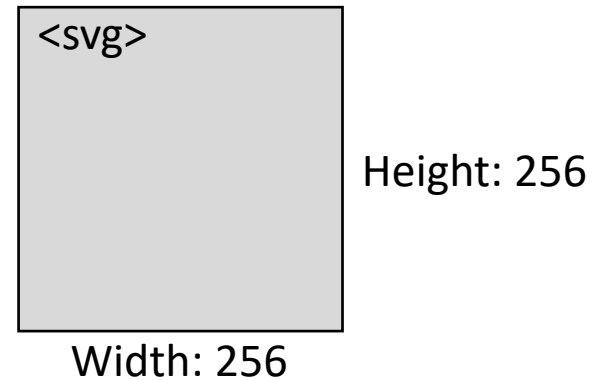
w06\_ex02\_main.js



# Example 02

- **Linear Scaling**

- SVG drawing region
  - Width: 256
  - Height: 256



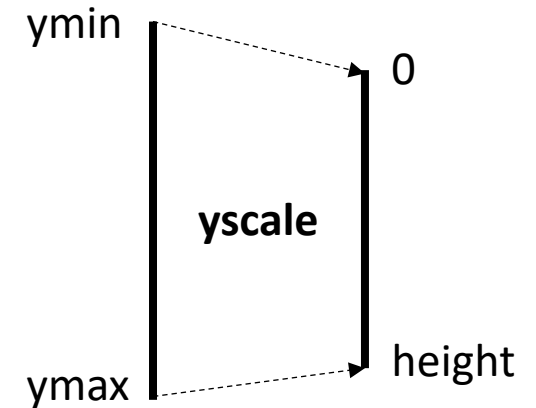
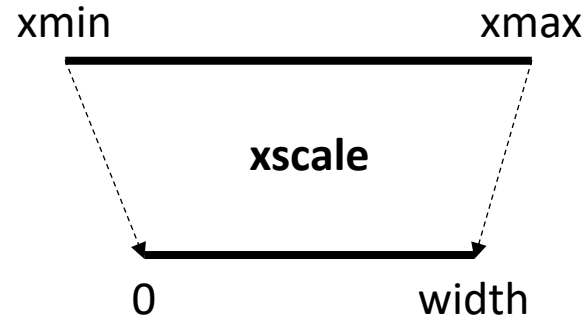
```
const width = 256;
const height = 256;
var svg = d3.select("body").append("svg")
    .attr('width', width)
    .attr('height', height);
```

w06\_ex02\_main.js

# Example 02

- **Linear Scaling**

- Scaling x and y axes



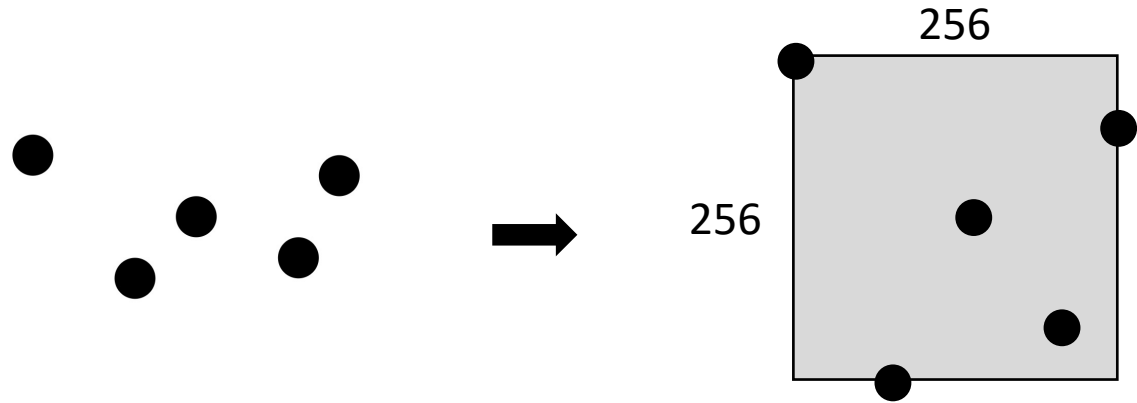
```
var xscale = d3.scaleLinear()  
  .domain( [d3.min(data, d => d.x), d3.max(data, d => d.x)] )  
  .range( [0, width] );  
  
var yscale = d3.scaleLinear()  
  .domain( [d3.min(data, d => d.y), d3.max(data, d => d.y)] )  
  .range( [0, height] );
```

w06\_ex02\_main.js

# Example 02

- **Linear Scaling**

- Draw circles

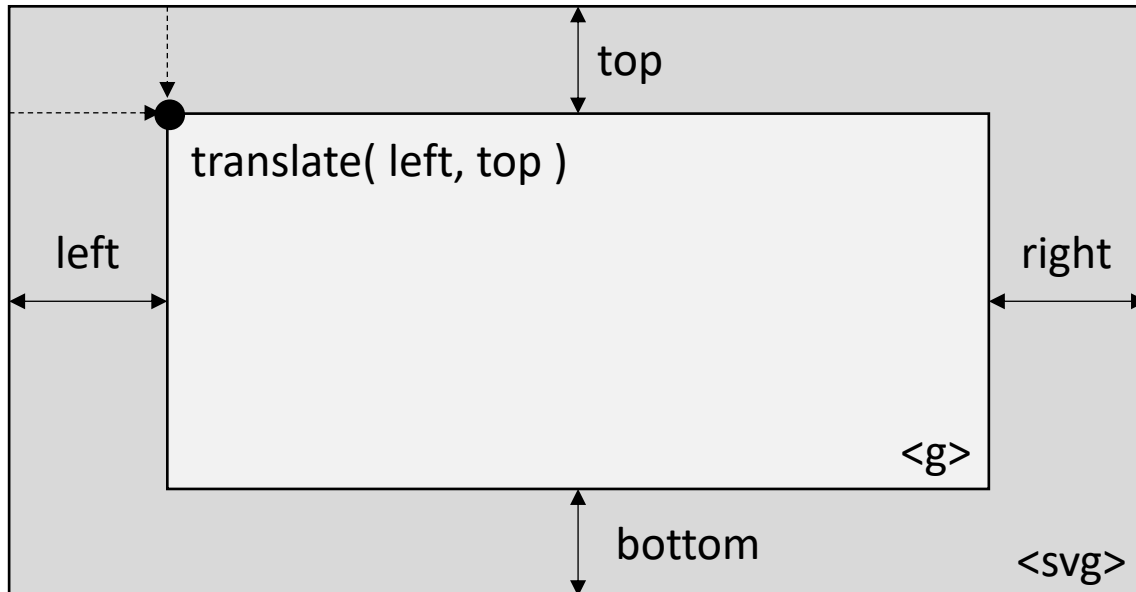


```
svg.selectAll("circle")  
  .data(data)  
  .enter()  
  .append("circle")  
  .attr("cx", d => xscale(d.x))  
  .attr("cy", d => yscale(d.y))  
  .attr("r", d => d.r);
```

w06\_ex02\_main.js

# Margin Convention

- Margins
  - Specified as an object
    - Top, right, bottom, left



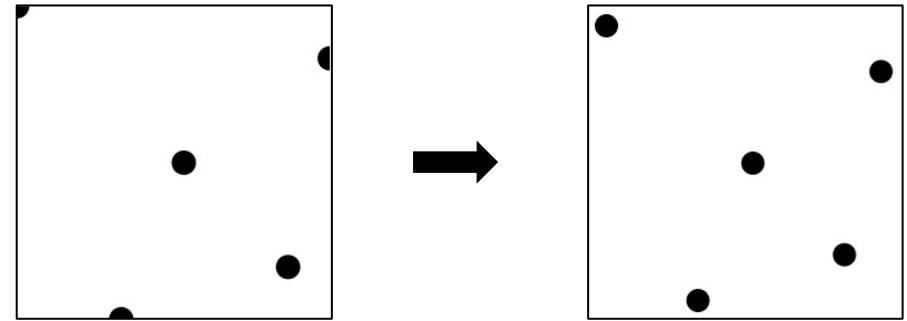
```
// Margins
const m = {
  top: x, right: x, bottom: x, left: x };

// 'svg' and 'g' elements
var svg = d3.select("body").append("svg")
  .attr('width', width)
  .attr('height', height);
  .append('g')
  .attr('transform', `translate(${m.left},
    ${m.top})`);
```

# Example 03

- **Margins**

- Draw circles inside the region specified with margins

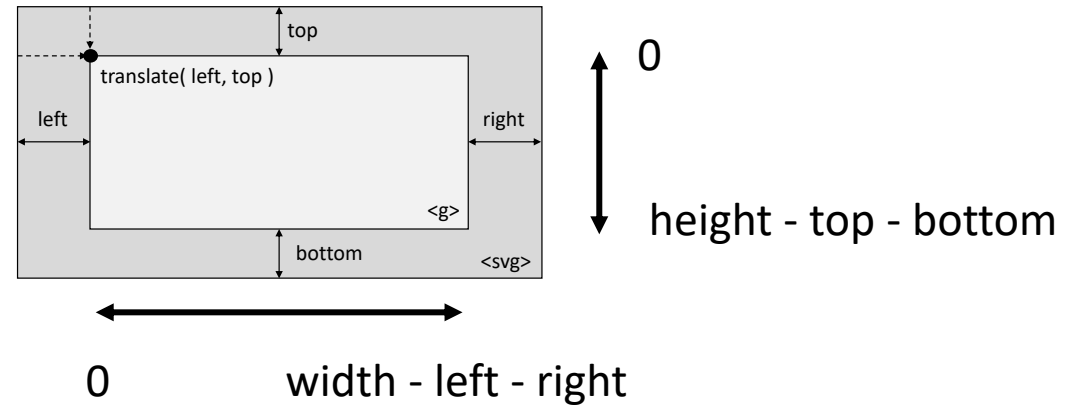


```
const width = 256;
const height = 256;
const margin = {top: 10, right: 10, bottom: 10, left: 10};
var svg = d3.select("body").append("svg")
  .attr('width', width)
  .attr('height', height)
  .append('g')
  .attr('transform', `translate(${margin.left}, ${margin.top})`);
```

w06\_ex03\_main.js

# Example 03

- **Margins**
  - Modify the scaling ranges



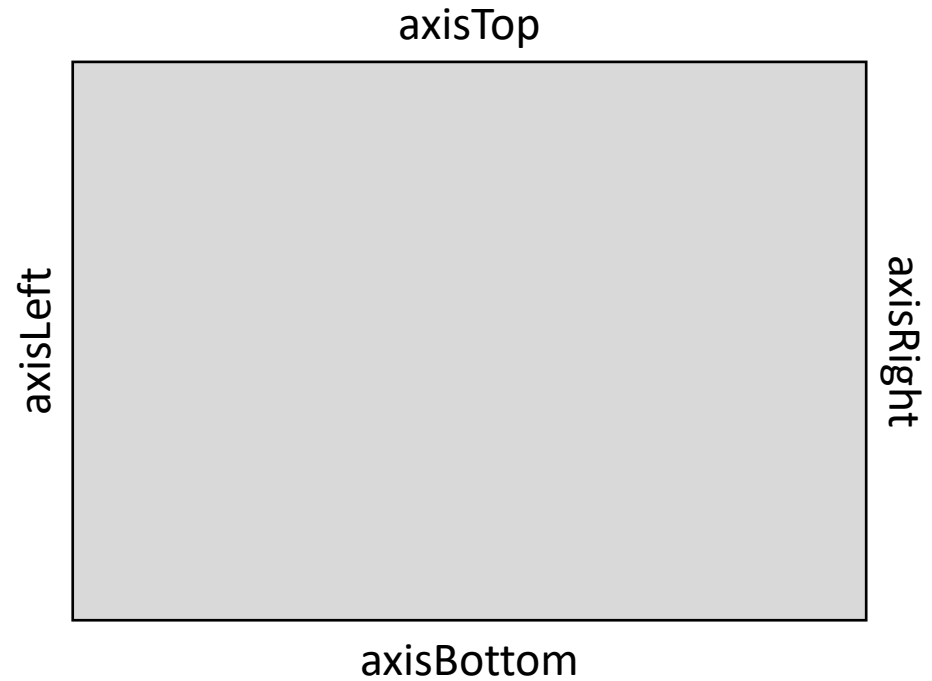
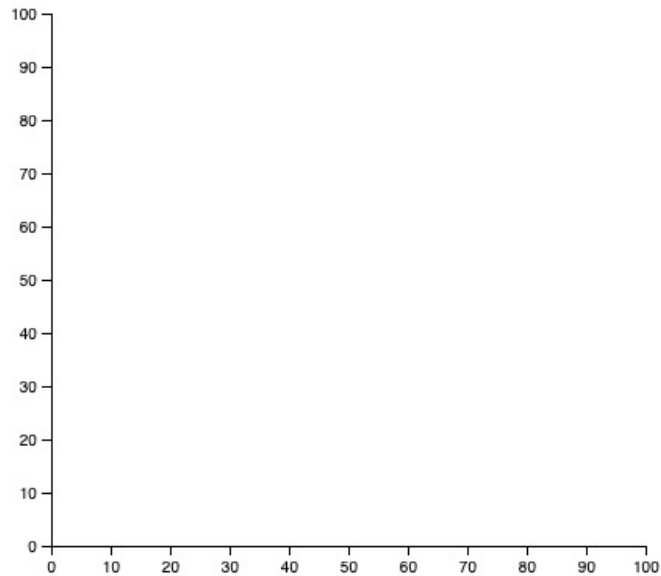
```
var xscale = d3.scaleLinear()  
  .domain( [d3.min(data, d => d.x), d3.max(data, d => d.x)] )  
  .range( [0, width - margin.left - margin.right] );  
  
var yscale = d3.scaleLinear()  
  .domain( [d3.min(data, d => d.y), d3.max(data, d => d.y)] )  
  .range( [0, height - margin.top - margin.bottom] );
```

w06\_ex03\_main.js

# Axis

- **Drawing Axes**

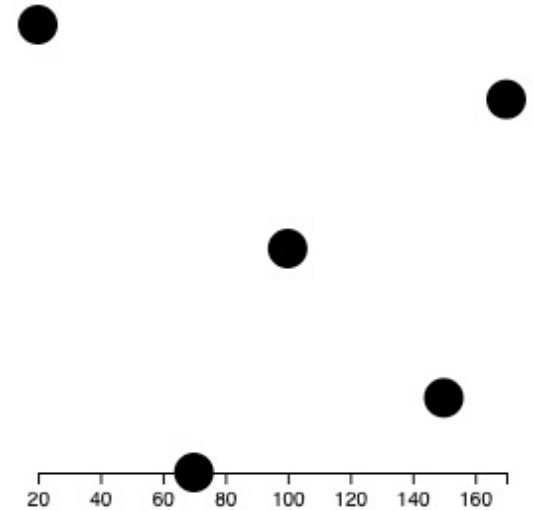
- d3.axisTop, d3.axisBottom, d3.axisLeft, d3.axisRight



# Example 04

- **Drawing Axes**

- Add a x-axis at the bottom of the region
- Modify the margin of the region



```
var xaxis = d3.axisBottom( xscale )  
    .ticks(6);
```

```
svg.append('g')  
    .attr('transform', `translate(0, ${height - margin.top - margin.bottom})` )  
    .call( xaxis );
```

w06\_ex04\_main.js

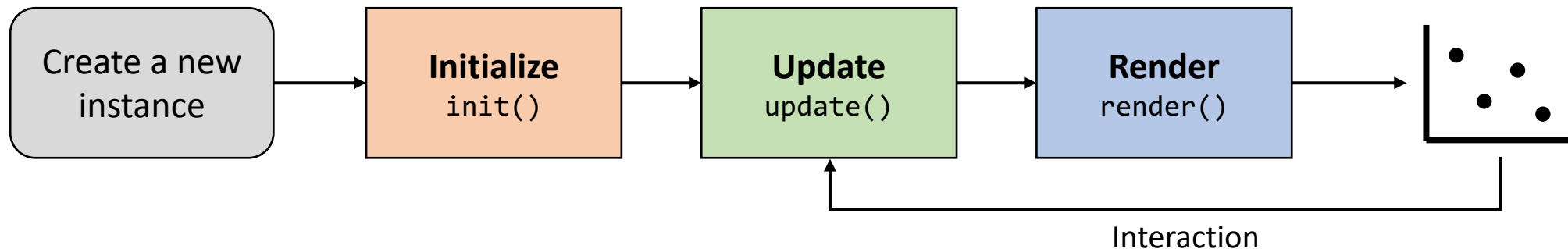


# Reusable Component

- **ScatterPlot Class**

- Visualization pipeline
  - **Initialize:** Create SVG elements and static components
  - **Update:** Update visual elements
  - **Render:** Map data to visual elements

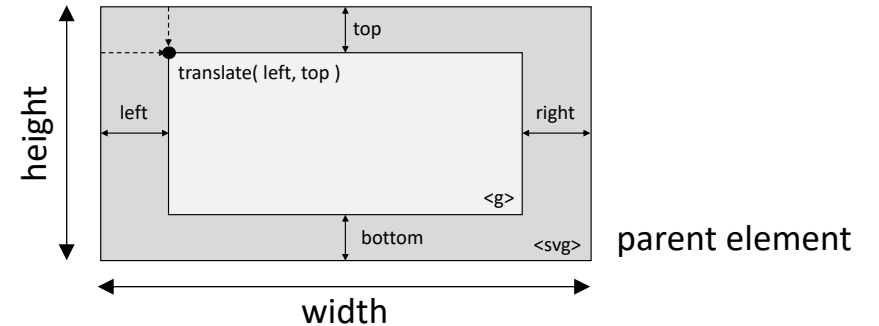
```
class ScatterPlot {  
    constructor() { ... }  
    init() { ... }  
    update() { ... }  
    render() { ... }  
    ...  
}
```



# Example 05

- **ScatterPlot Class**

- **constructor( config, data )**
  - config = {parent, width, height, margin}
  - margin = {top, right, bottom, left}



```
constructor( config, data ) {  
  this.config = {  
    parent: config.parent,  
    width: config.width || 256,  
    height: config.height || 256,  
    margin: config.margin || {top:10, right:10, bottom:10, left:10}  
  }  
  this.data = data;  
  this.init();  
}
```

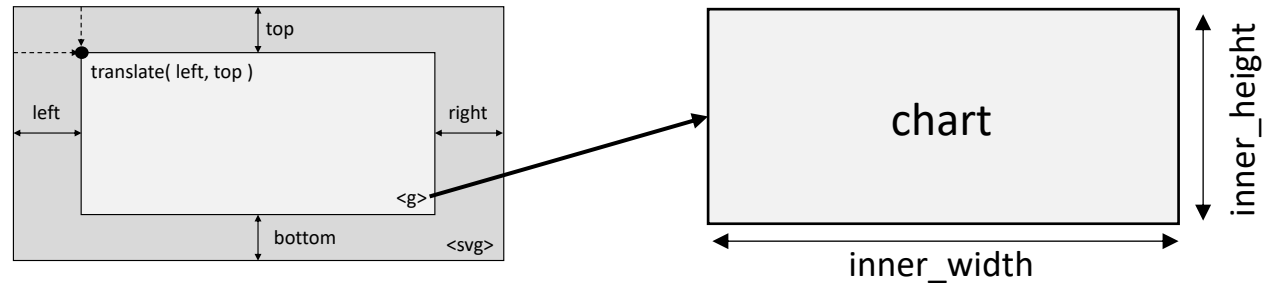
w06\_ex05\_main.js

# Example 05

- **ScatterPlot Class**

- **init()**

- Initialize drawing region
      - svg and chart



```
init() {  
  let self = this;  
  
  self.svg = d3.select( self.config.parent )  
    .attr('width', self.config.width)  
    .attr('height', self.config.height);  
  
  self.chart = self.svg.append('g')  
    .attr('transform',  
      `translate(${self.config.margin.left}, ${self.config.margin.top})`);  
  
  self.inner_width = self.config.width - self.config.margin.left - self.config.margin.right;  
  self.inner_height = self.config.height - self.config.margin.top - self.config.margin.bottom;  
  ...  
}
```

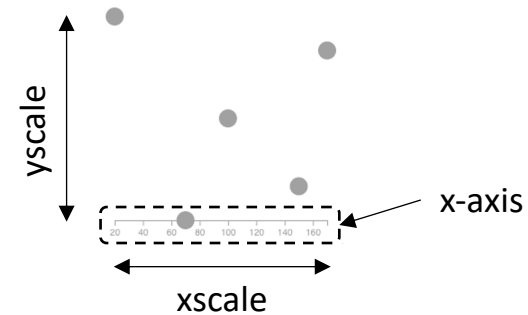
w06\_ex05\_main.js

# Example 05

- **ScatterPlot Class**

- **init()**

- Initialize scales and x-axis
      - xscale, yscale, xaxis, xaxis\_group



```
...
self.xscale = d3.scaleLinear()
    .range( [0, self.inner_width] );

self.yscale = d3.scaleLinear()
    .range( [0, self.inner_height] );

self.xaxis = d3.axisBottom( self.xscale )
    .ticks(6);

self.xaxis_group = self.chart.append('g')
    .attr('transform', `translate(0, ${self.inner_height})`);
}
```

w06\_ex05\_main.js

# Example 05

- **ScatterPlot Class**
  - **update()**
    - Update scaling domains
    - Call render()

```
update() {  
  let self = this;  
  
  const xmin = d3.min( self.data, d => d.x );  
  const xmax = d3.max( self.data, d => d.x );  
  self.xscale.domain( [xmin, xmax] );  
  
  const ymin = d3.min( self.data, d => d.y );  
  const ymax = d3.max( self.data, d => d.y );  
  self.yscale.domain( [ymin, ymax] );  
  
  self.render();  
}
```

w06\_ex05\_main.js

# Example 05

- **ScatterPlot Class**

- **render()**
  - Draw circles
  - Draw x-axis

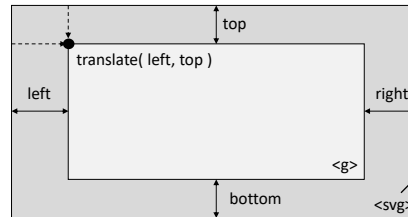
```
render() {  
  let self = this;  
  
  self.chart.selectAll("circle")  
    .data(self.data)  
    .enter()  
    .append("circle")  
    .attr("cx", d => self.xscale( d.x ) )  
    .attr("cy", d => self.yscale( d.y ) )  
    .attr("r", d => d.r );  
  
  self.xaxis_group  
    .call( self.xaxis );  
}
```

w06\_ex05\_main.js

# Example 05

- **Loading data**

- Create and initialize scatterplot
- Show scatterplot



`<svg id="drawing_region"></svg>`

```
d3.csv("https://xxx.github.io/InfoVis2022/W04/data.csv")
  .then( data => {
    data.forEach( d => { d.x = +d.x; d.y = +d.y; });

    var config = {
      parent: '#drawing_region',
      width: 256,
      height: 256,
      margin: {top:10, right:10, bottom:20, left:10} };

    const scatter_plot = new ScatterPlot( config, data );
    scatter_plot.update();
  })
  .catch( error => { console.log( error ); } );
```

w06\_ex05\_main.js

# Example 05

- **HTML**

- Add empty SVG drawing region
- Run w06\_ex05\_main.js

```
<html>
  <head>
    <title>W06: Example 05</title>
  </head>
  <body>
    <svg id="drawing_region"></svg>
    <script src="https://d3js.org/d3.v6.min.js"></script>
    <script src="w06_ex05_main.js"></script>
  </body>
</html>
```

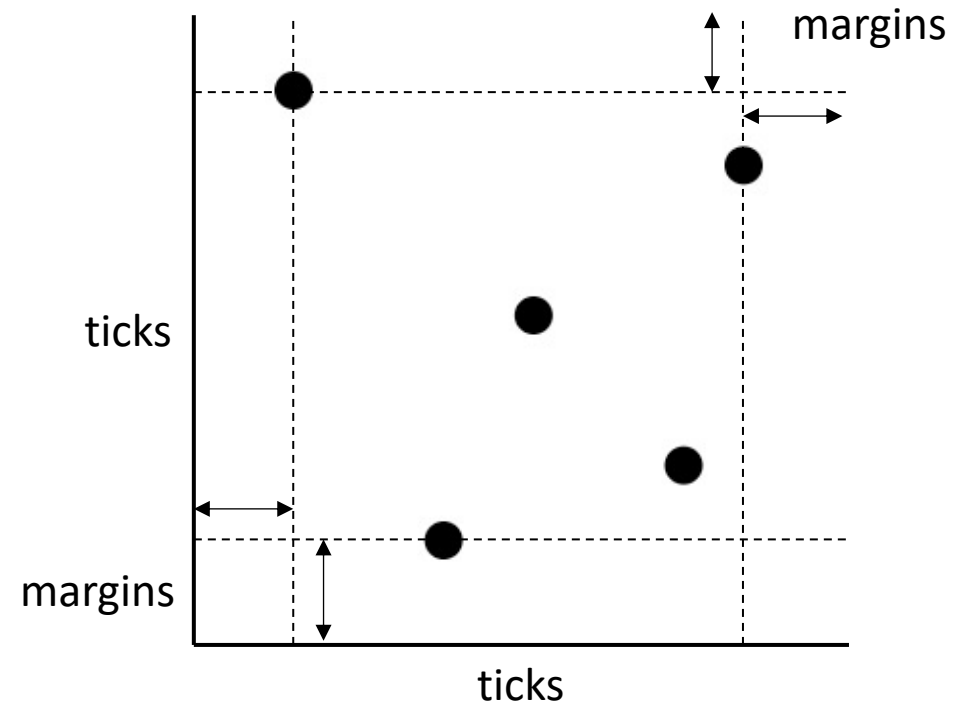
w06\_ex05\_index.html



# Task 1

- **Add y-axis to Example 05**

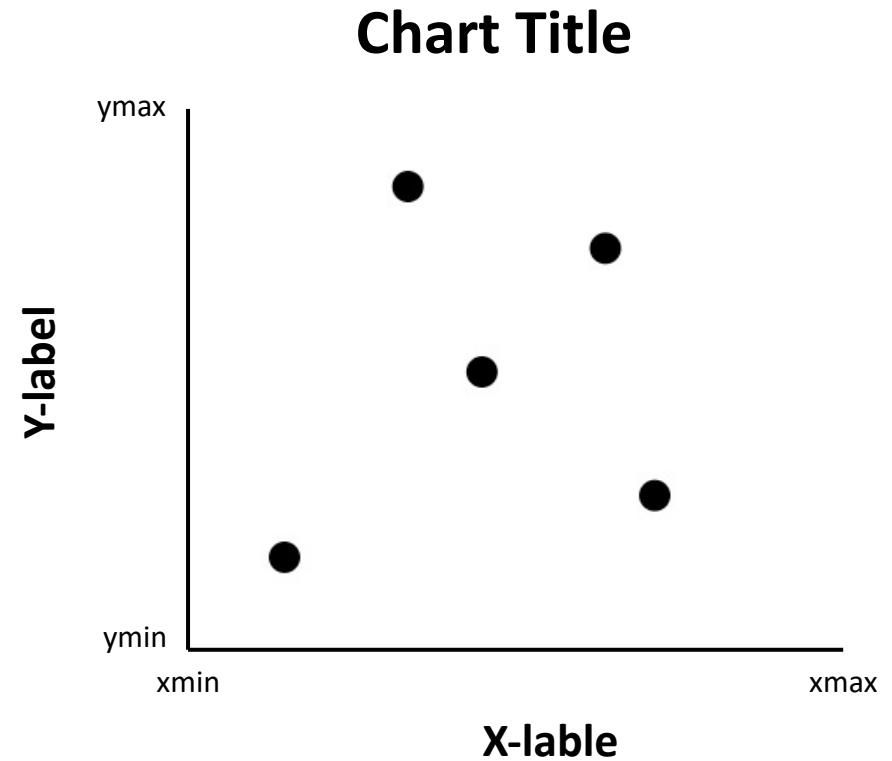
- Load an external data file
  - Same data as W04 - Task1
  - Other new data
  - ...
- Add margins to each axis
- Modify ticks for each axis
  - ticks
  - tickSize
  - tickPadding
  - ...



# Task 2

- **Add chart title and axis labels to Task 1**

- Set origin to left-bottom
- Modify font size and weight



# Polling

- **Take the poll**
  - Student ID Number
  - Name
  - URL to Task 1
  - URL to Task 2
- **Submission deadline**
  - **May 9 (Mon), 2022 by 23:59 JST**