

Piotr Kaczmarczyk

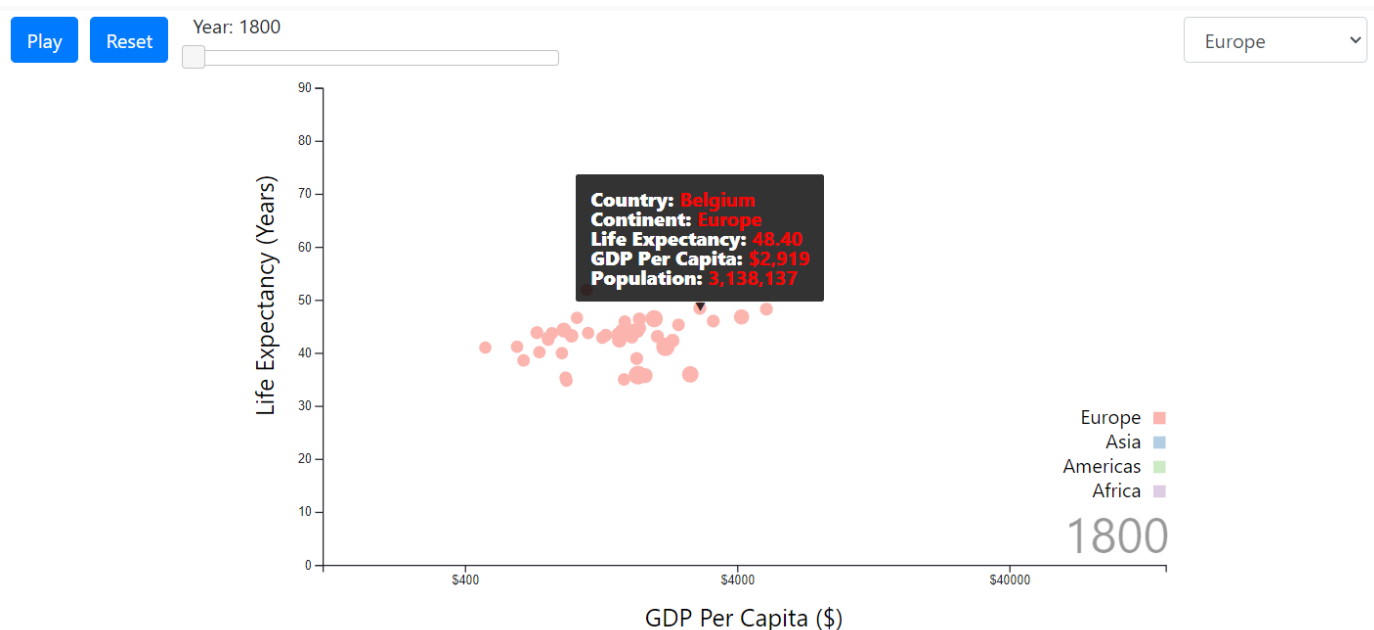
Symulacja komputerowa – projekt

**Temat: Symulacja przebiegu oczekiwanej długości życia względem produktu krajowego brutto.**



Wykorzystałem w projekcie głównie HTML, JS oraz Bootstrap. Aby generować wykresy użyłem D3.js oraz jQuery.

W górnym panelu możemy uruchomić symulację oraz kontrolować oś czasu. Po prawej stronie widzimy listę, z której dostępne są opcje filtrowania danych.



Jak widać powyżej po najechaniu na konkretną kropkę widzimy okienko z informacjami na temat poszczególnego obiektu.

W prawym dolnym rogu znajduje się legenda oraz pokazany mamy obecnie wybrany rok.

Poniżej widzimy panel do obsługi symulacji, możemy z niego wybrać parametry symulacji oraz ją zrestartować.

Simulate Reset data

Multiply value

1,1

Parameter to multiply:

All

Continent:

All

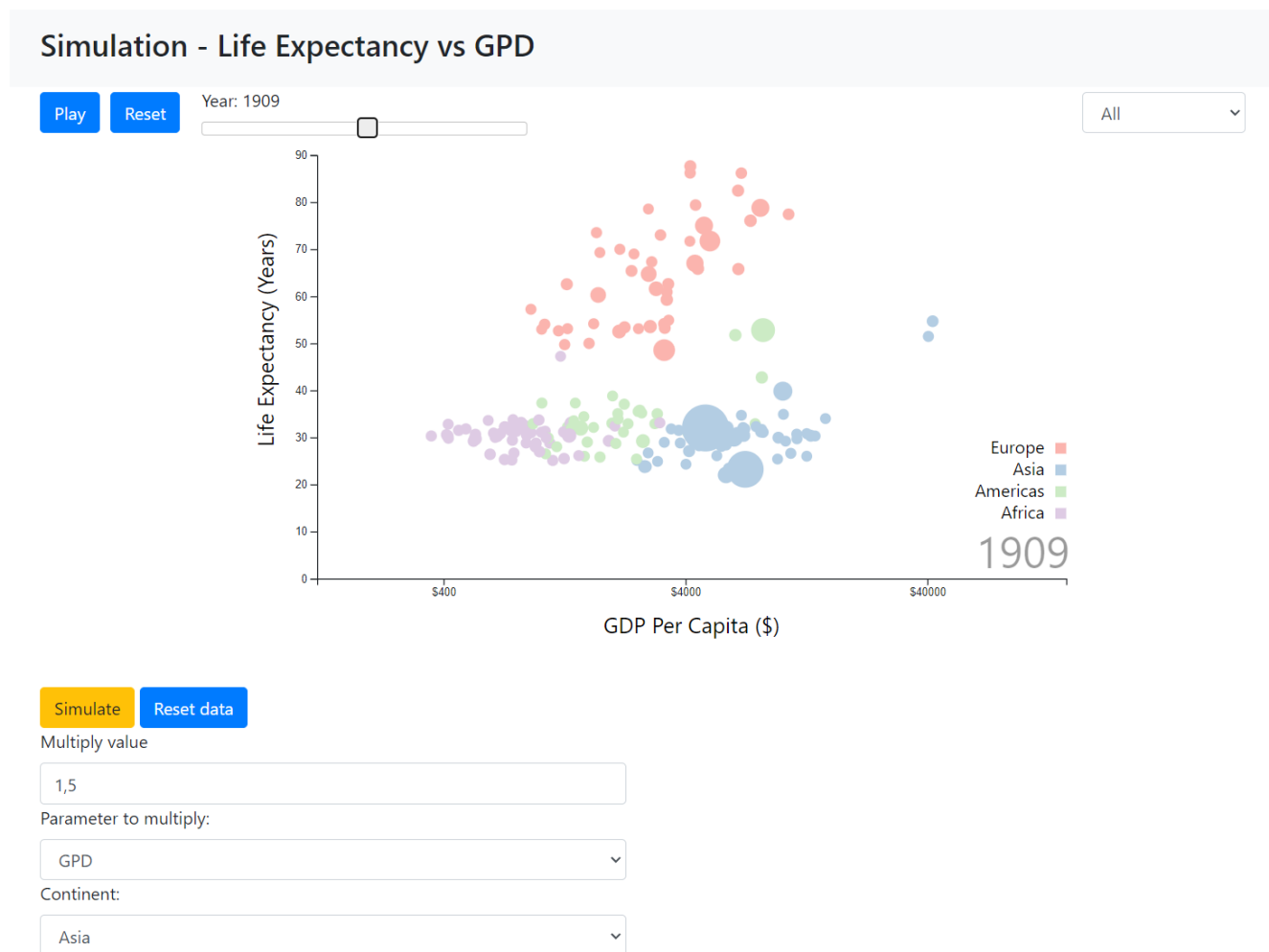
Dzięki przyciskom możemy generować naszą symulację albo ją resetować wczytując początkowe dane.

Parametrami jakie możemy ustawić to:

- Mnożnik
- Parametr do zmiany
- Kontynent

**Rozkład liczb losowych** został zaimplementowany w parametrze “Multiply value”, w którym możemy ustawić jak zostaną przemnożone wartości symulacji.

Przykładowe ustawienie symulacji oraz jej widok po uruchomieniu przebiegu.



Funkcje obsługujące symulacje zostały umieszczone w pliku "multiply.js" prezentują się one następująco:

```
const max_year = 214;
```

```

function multiply() {
  const multiValue = document.getElementById("multiply").value
  const continent = document.getElementById("continent-multiply").value
  const field = document.getElementById("field-multiply").value //income, life_exp

  for (let year = 0; year < max_year; year++)
    for (let country = 0; country < formattedData[year].length; country++) {
      if (continent === 'all')
        if (field !== 'all'){
          formattedData[year][country][field] *= parseFloat(multiValue)
        }
      else {
        formattedData[year][country]['income'] *= parseFloat(multiValue)
        formattedData[year][country]['life_exp'] *= parseFloat(multiValue)
      }
      else if (formattedData[year][country]['continent'] === continent)
        if (field !== 'all'){
          formattedData[year][country][field] *= parseFloat(multiValue)
        }
      else {
        formattedData[year][country]['income'] *= parseFloat(multiValue)
        formattedData[year][country]['life_exp'] *= parseFloat(multiValue)
      }
    }
  update(formattedData[time])
}

function reset() {
  load_data()
  $("#year")[0].innerHTML = 1800
  $("#date-slider").slider("value", Number(1800))
  time = 0
}

```

Obsługa wykresu została opisana w pliku "main.js". Najważniejsza w nim jest funkcja "update(data)", która pozwala nam manipulować wykresem i zmieniać jego wygląd w zależności od ustawionych parametrów. Wygląda ona następująco:

```
function update(data) {
  // standard transition time for the visualization
  const t = d3.transition()
    .duration(100)

  const continent = $("#continent-select").val()

  const filteredData = data.filter(d => {
    if (continent === "all") return true
    else {
      return d.continent == continent
    }
  })

  // JOIN new data with old elements.
  const circles = g.selectAll("circle")
    .data(filteredData, d => d.country)

  // EXIT old elements not present in new data.
  circles.exit().remove()

  // ENTER new elements present in new data.
  circles.enter().append("circle")
    .attr("fill", d => continentColor(d.continent))
    .on("mouseover", tip.show)
    .on("mouseout", tip.hide)
    .merge(circles)
    .transition(t)
    .attr("cy", d => y(d.life_exp))
    .attr("cx", d => x(d.income))
    .attr("r", d => Math.sqrt(area(d.population) / Math.PI))

  // update the time label
  timeLabel.text(String(time + 1800))

  $("#year")[0].innerHTML = String(time + 1800)
  $("#date-slider").slider("value", Number(time + 1800))
}
```

Wnioski:

Tworząc ten projekt mogłem poznać bibliotekę "d3.js" oraz wykorzystać ją w praktyce. Pozwoliło to stworzyć mi interaktywną symulację przedstawiającą to jak możemy manipulować danymi i je pokazywać w praktyce.

