Uživatelská dokumentace k samostatné práci

2

z předmětu KIV/UPG

Jméno a příjmení: Lukáš Černý

Datum: 1. 4. 2014

Osobní číslo: A13B0286P

Vstupní data

Vstupní data jsou parsoval do formátu JSON. Tímto jsem zajistil snadnou přenositelnost do jiných programovacích jazyků (např. javascript). Standartně tento formát java nepodporuje, je potřeba tedy použít externí knihovnu. Já použil knihovnu *google-gson-2.2.4*.

Snadnou rozšířitelnost pro další jsem zajistil tím, že je k programu vytvořen parsér, který ze souboru TXT vytvoří JSON, se kterým pak program pracuje. Jelikož jsou I/O operace nejnáročnější na čas, načítám pouze jeden soubor. Soubory TXT musí mít název *'rok'.txt* a daný rok uložený v souboru *years.txt*. Data v souboru TXT musí být v požadovaném formátu (viz. níže).

Nepřesnosti v datech

V roce 2004 si myslím, že jsou špatně uvedené hodnoty pro V. onemocnění. V jiných letech se čísla na řádce A pohybují v tisících, zde se pohybují v desetitisících a podobné je to i na řádce B, kde se čísla v jiných letech pohybují ve stovkách a zde v tisících. Upravil jsem hodnoty, aby přibližně odpovídali hodnotám v jiných letech. Když bych tyto data neupravil, tak takto vysoká čísla by dělala problém v aplikaci.

Formát dat v TXT souboru

PHA PLZ ... CELKEM

I. a hodnota, hodnota,..., hodnota

I. b hodnota, hodnota,..., hodnota

II a hodnota, hodnota,..., hodnota

.

.

.

XXI. a hodnota, hodnota,..., hodnota

XXI. b hodnota, hodnota,..., hodnota

Formát dat v JSON souboru

stránka č.3.

3

```
wrap {
  2000 {
    PHA {
          a [I, II, III, .., XXI]
      b [I, II, III, .., XXI]
    },
    STC {
          a [I, II, III, .., XXI]
          b [I, II, III, .., XXI]
    }
    .
    .
    CELKEM {
          a [I, II, III, .., XXI]
          b [I, II, III, .., XXI]
    }

  },
  2001 {
    PHA {
          a [I, II, III, .., XXI]
          b [I, II, III, .., XXI]
    },
    STC {
          a [I, II, III, .., XXI]
          b [I, II, III, .., XXI]
    }
    .
    .
    CELKEM {
          a [I, II, III, .., XXI]
          b [I, II, III, .., XXI]
    }
  }
  .
  .
  2012 {
    PHA {
          a [I, II, III, .., XXI]
          b [I, II, III, .., XXI]
    },
    STC {
          a [I, II, III, .., XXI]
          b [I, II, III, .., XXI]
    }
    .
    .
    CELKEM {
          a [I, II, III, .., XXI]
          b [I, II, III, .., XXI]
    }
  }
}
```

GUI

Barevná škála

Barevnou škálu pro detailní vizualizaci dat jsem rozdělil do 4 kategorií. Po přepočítání nejmenší a největší hodnoty v daném roce, je hodnota pro daný region zařazena na základě toho, do jaké ¼ patří. Podle výsledku je přiřazena barva (viz. legenda). Na podobném principu je udělán i výpočet trendu a následné zvolení barvy.

Souhrnná vizualizace

Souhrnná vizualizace se nachází za posledním rokem. Spustíme-li animaci, na tom to snímku bude čekací doba 5000ms. Při spuštění lze souhrnnou vizualizaci zobrazit parametrem *-1*.

Tlačítka

Aplikace se ovládá pomocí 4 tlačítek.

|< = zobrazí první snímek (rok)

>| = zobrazí poslední snímek (rok nebo souhrnnou vizualizaci)

< = zobrazí předchozí rok (zastaví se na 1. roce)

> = zobrazí další rok (zastaví se na posledním roce nebo na souhrnné vizualizaci)

Run/Pause = spustí/zastaví animaci, kdy se jednotlivé roky mění po 1250ms a souhrnná vizualizace se zobrazí na 5000ms

Export do SVG

Z důvodu špatného návrhu mé semestrální práce nebylo možné udělat export do formátu SVG. Jednotlivé roky ukládám jako BufferImage při spuštění aplikace. Po zjištění, že z BufferImage export do SVG neudělám, musel jsem tento krok vynechat. Chyby jsem si nevšiml, protože export jsem chtěl dělat až na konci semestrální práce.

Struktura programu

Myslím si, že celková struktura programu by mohla být lépe navržená. Jelikož na semestrální práci nebylo moc času, přidával jsem jednotlivé části kódu až v době, kdy jsem je potřeboval. Například kód pro vizualizaci trendu by mohl být oddělen od kódu pro detailní vizualizaci.