Extract Popularity Time Series

http://stats.grok.se/json/en/

Extract and Print on a jason file

Create a list of dates as strings “200910” - “YearMonth” - giving the dates Start and End, in this form {2010,10} (as list of numbers).

```
createDates[init_,end_]:= StringJoin[ToString[#],
    If[#2<10,StringJoin[ToString[0],ToString[#2]],ToString[#2]]]&@@@ DateRange[init,en

(* input of this form {2010,10} (as list of numbers),
 output has this form {"200910","200911","200910",..} *)
```

```
importingOnFile[dates_List,pages_,outputfile_]:=
Do[
    (Import["http://stats.grok.se/json/en/"<>#1<>"/"<>pages[[i]], "JSON"]>>>
        "/Users/Levantina/Documents/FISICA/TESIPOP/Timeseries/"<>outputfile) & /@ date
    ,
    {i,Length[pages]}]

(* dates_List = {"200910","200911","200910",..} in months, in crescent order, pages =
```

```
importingOnFile2[dates_List,pages_,outputfile_]:=
Do[
    (Import["http://stats-classic.grok.se/json/en/"<>#1<>"/"<>pages[[i]], "JSON"]>>>
        "/Users/Levantina/Documents/FISICA/TESIPOP/Timeseries/"<>outputfile) & /@ date
    ,
    {i,Length[pages]}]

(* dates_List = {"200910","200911","200910",..} in months, in crescent order, pages =
```

Clean Time Series

This function reads from file the time series, knowing how many months.

```
readingTimeSeries[file_,Nmonths_]:= ReadList["/Users/Levantina/Documents/FISICA/TESIPO
```

```
CleanTimeSeries[series_]:={FromDigits /@ StringSplit[#,"-"],#2}& @@@ series[[1,2]]
```

This function cleans imported data and makes them ready to be plotted and analyzed.

```
ExtractTimeSeries[imported_List,Nmonths_Integer]:=Partition[CleanTimeSeries[#]& /@ imp
```

Process and Plot Cleaned Time Series

This function is useful to study the average behavior of a sample of pages.

Mean and Standard Deviation of visitors of a page in time.

```
SafeMean[onePage_]:= N@Mean[Cases[Flatten @ onePage[[All,All,2]],Except[0]]]
SafeStandardDev[onePage_]:= N@StandardDeviation[Cases[Flatten @ onePage[[All,All,2]],E
```

```
averageTimeSeries[cleanedTS_]:= N@Mean[Flatten[cleanedTS[[#,All,All,2]]]& /@ Range[1,L
```

This function plots the popularity for the selected Vertex:

```
PlotTimeSeries[series_,opts___]:= DateListPlot[series,Joined →True,opts]
```

```
ShowTimeSeries[index_,vertices_,imported_List,Nmonths_Integer]:=
    PlotTimeSeries[
        ExtractTimeSeries[imported,Nmonths][[index]],
            PlotRange→All,PlotLabel→vertices[[index]]
    ]
```

Remove non - existing days from Time Series:

```
LeapYearQ[2010]
```

```
False
```

```
commonYear = {31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};

leapYear = {31, 29, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};

nonExistingCommonDays = {{{__, 02, 30}, __}, {{__, 2, 31}, __},
   {{__, 4, 31}, __}, {{__, 06, 31}, __}, {{__, 09, 31}, __}, {{__, 11, 31}, __}};

nonExistingLeapDays = {{{__, 02, 29}, __}, {{__, 02, 30}, __}, {{__, 2, 31}, __},
   {{__, 4, 31}, __}, {{__, 06, 31}, __}, {{__, 09, 31}, __}, {{__, 11, 31}, __}};
```

```
RelativeDistance[onePage_]:=With[{m=N@Mean[Cases[Flatten @ #,Except[0]]]},(If[#!=0,(N@
```

```
veryGoodTSB =
  DeleteCases[#, Alternatives @@ If[LeapYearQ[#[[1, 1, 1]]], nonExistingLeapDays,
         nonExistingCommonDays]] & /@ # & /@ goodTSB;

veryGoodTSA =
  DeleteCases[#, Alternatives @@ If[LeapYearQ[#[[1, 1, 1]]], nonExistingLeapDays,
         nonExistingCommonDays]] & /@ # & /@ goodTSA;
```