SRA GROUP
GSLetterNeo Vol.116
2018年3月

時系列データをプログラムで操作して見る

松原 伸人
matubara@sra.co.jp

◆時系列データを操作するプログラム

時間情報を持つ大量のデータを見て、データの特性を探索するようなツールの開発を行っています。

時間情報を持つデータは、テキストを書いた時刻、ニュース記事を読んだ時刻、写真を撮った時刻、絵を描いた時刻、ビデオを撮影した時刻などを扱っています。

現在、chronicle.js という名前で Web ブラウザ上で動作するプログラムを JavaScript で記述しています。

Vol.102 と 103 で紹介したように、chronicle.js は、何時何が起こったかを表すイベント群を表すタイムラインと、何時から何時までが何かを表すルーラーと、時間の区切りを表すタイムセパレーターと、数値で表される出来事群を扱うグラフを使って、データを作ることができるようになっています。

chronicle.js は、これらにより記述されたデータファイルを読み込んで、一連のデータを操作できるようにします。

GSLetterNeo 年表は、発行した記録のタイムラインと、関連しそうな記事のタイムラインと、各号に含まれる語の数のグラフと、西暦を表すルーラーと、和暦を表すルーラーと、発行者を表すルーラーと、編集者を表すルーラーと、時代を表すルーラーで構成しています。

GSLetterNeo 年表を記述したデータファイルは gsletterneochronicle.md です。

このデータファイルの文字コードは UTF-8 です。

タイムラインとルーラーとグラフは、データファイルに書いた順に表示されるようになっています。

GSLetterNeo をブラウザの Web インスペクタで開くとデータファイルを読み込むプログラムが読めます。

ここでは、chronicle.js とデータファイル gsletterneochronicle.md が同じ sra.co.jp の web サーバにあるため、XMLHttpRequest を用いてファイルを text として読み込んでいます。

FileListLoader は XMLHttpRequest を使って複数のファイルを読み込むプログラムです。

Chronile オブジェクトを作って parse 命令で読み込んだテキストを指定してデータを読み込んでいます。

```
var chronicle = new Chronicle().parse(text)
```

ChronicleComponentVertically.js は Chronicle オブジェクトを HTML で縦書きの年表として表示します。

```
var chronicleComponent = new ChronicleComponentVertically(chronicle).appendTo(document.getElementById('chronicleArea'))
```

ChronicleControl.js は、出来事を順に見ていくボタンと、出来事のテキストよる検索で構成するユーザインターフェイスです。

縦書き年表のオブジェクト chronicleComponent を指定してオブジェクトを作ると、順に見ていくボタン操作と検索操作が縦書き年表に加わります。

```
new ChronicleControl(chronicleComponent).appendTo(document.getElementById('chronicleControl))
```

◆出来事のテキスト検索

「アジャイル」を含む出来事を検索するには findEvents ByText を使います。

検索結果は出来事順の配列で得られます。

以降のコードは、GSLetterNeo - inhouse - KTL をブラウザの Web インスペクタで開いて実行できるようになっています。
Apple Safari の場合、上記の URL を開いて、メニューの「開発」の「JavaScript コンソールを表示」でコンソールを開き、コンソールの最下部にあるコード入力欄に、記事内のコードをコピーアンドペーストして実行できます。
Google Chrome の場合、メニューの「その他のツール」の「デベロッパーツール」で Console タブをクリックして開いた画面で実行できます。

```
chronicle.findEventsByText('アジャイル')
```

「アジャイル」を含む記事の発行日時を得るには Event オブジェクトの startDate 関数を用います。

```
chronicle.findEventsByText('アジャイル').map(function (foundEvent) {
	return (foundEvent instanceof Chronicle.Event) && foundEvent.startDate()
})
```

◆出来事を見る

Chronicle オブジェクトは、全出来事を時間順にソートした配列 events を持っています。

例えば最初に起きた出来事、最後に起きた出来事は、次のように events 配列を参照して得られます。

```
chronicle.events()[0]
chronicle.events()[chronicle.events().length - 1]
```

「アジャイル」で検索して見つけた出来事を区切りにして、出来事を時間順に次々見ていくような場合、次のようにかけます。

```
var foundEvents = chronicle.findEventsByText('アジャイル').filter(function (event) {
	return event instanceof Chronicle.Event;
}),
results = [];
for (var foundIndex = 0; foundIndex < foundEvents.length - 1; foundIndex += 1) {
	var foundEvent = foundEvents[foundIndex],
		nextEvent = foundEvents[foundIndex + 1],
		startIndex = chronicle.events().indexOf(foundEvent) + 1,
		endIndex = chronicle.events().indexOf(nextEvent) - 1,
		result = [];
	for (var index = startIndex; index <= endIndex; index += 1) {
		result.push(chronicle.events()[index])
	}
	results.push(result)
}
results
```

◆時間にそってデータを見る

Chronicle オブジェクトはデータファイルを parse すると、タイムラインごとに、同時刻に起きた出来事の集合を Times オブジェクトとして保持し、Times の時刻による索引_timesByDate を生成します。

データの中に出てくる時刻の一覧は、timeStrings 関数で得られます。

```
chronicle.timeStrings()
```

時刻ごとに全出来事を見ていく場合、timesDo 関数を用いるか、timeStrings と timesByDate を用います。

```
chronicle.timesDo(function (timeString, timesArray, index) {
	console.log(timeString, timesArray);
})
```

timesArray は、times オブジェクトのリストです。タイムラインごとの Times オブジェクトが入っています。

```
chronicle.timeStrings().forEach(function (timeString) {
	var timesArray = chronicle.timesByDate()[timeString]
	console.log(timeString)
	timesArray.forEach(function (times) {
		times.events().forEach(function (event) {
			console.log(event);
		})
	})
})
```

◆時間で出来事を検索する

findTimesWithDate 関数を使うと指定した時刻の出来事を持つ Times オブジェクトが得られます。 例えば Vol100 が発行された 2016 年 11 月頃の出来事を検索する場合次のようにかけます。

```
chronicle.findTimesWithDate(new Date('2016-11-01'))
```

この関数は指定された日付に最も近い時刻の出来事を見つけるようになっています。

◆出来事を軸にしてデータを見る

datarepeat は試作中の時系列データの可視化方法です。

時系列データを読み込んだ Chronicle オブジェクトと、軸にする出来事群により、出来事群を画面上に等間隔の目盛とする軸として置き、目盛間の時刻に応じた位置に全データを配置します。

検索して見つけた出来事群を軸にしてデータ全体を見たり、カテゴリが同じ出来事群を軸にデータを見たりできます。

重要な出来事群を抜き出して軸にすると、その前後でのデータの量とか種類の違いが見えるような可視化方法を目標にしています。

この URL を開くと gsletterneochronicle-publish_timeline.md を読みこんだ Chronicle オブジェクトを指定して、ChronicleDatarepeat を作成します。

gsletterneochronicle-publish_timeline.md には、GSLetterNeo が発行された日、タイトル、著者、tfidf で抽出した名詞のうち数値が高い上位 10 単語が書かれています。

ChronicleDatarepeat は時系列データを HTML の Canvas で 2 次元描画します。

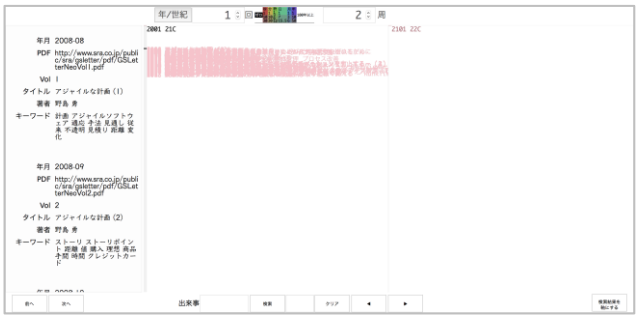

図 1 Datarepeat を開いた直後の画面

Datarepeat の詳細は次号で紹介します。

GSLetterNeo Vol.116
2018 年 3 月 20 日発行
発行者●株式会社 SRA 先端技術研究所
編集者●土屋正人

バックナンバを公開しています●http://www.sra.co.jp/gsletter
ご感想・お問い合わせはこちらへお願いします●gsneo@sra.co.jp

株式会社SRA
〒171-8513 東京都豊島区南池袋２－３２－８

夢を。Yawaraka Innovation
やわらかいのべーしょん