Milestone 3.3.3 – Visualisation Framework API Specification

This Document (M3.3.3) specifies the API on a technical level. It describes the internal (JavaScript), as well as the external HTTP based API.

Distribution

Version	Date of sending	Name	Role in project
0.2	09.08.2010	Vassilis Tzouvaras	Developer NTUA
0.2	09.09.2010	Christian Mahnke	Developer UGOE

Approval

Version	Date of approval	Name	Role in project
0.3	12.10.2010	Vassilis Tzouvaras	Developer

Revisions

Version	Status	Author	Date	Changes
0.1	Draft	UGOE – Christian Mahnke	01.07.10	Initial Version
0.2	Draft	UGOE – Christian Mahnke	15.07.10	• Regenerated links (they seem to work now)
0.3	Draft	UGOE – Christian Mahnke	07.10.10	Enhanced the Rationale section as proposed by Vassilis Tzouvaras.
1.0	Final	VPZ	12.10.2010	Final Layout

Table of Contents

Rationale

Scope and purpose of this document

As discussed with the programmers at The Hague (EDL) the purpose of this document is a API documentation to ensure maintainability after the end of Europeana. Connect. To make sure this goal is meet this document includes also the internal JavaScript API. A subset of this API will be later (as part of the second prototype) also be available as GWT based API.

The second part of the document contains the important part in terms of reusability, the HTTP based interaction with external systems.

How to use this APIs

JavaScript

The JavaScript based API can be used to configure and enhance the user experience from within a HTML page, another important use of the JavaScript based API is the use for a GWT Wrapper, as it will be part of the second prototype.

The easiest method of integrating the interface is a JavaScript section in the header of an HTML document, which initializes the interface:

```
<script type="text/javascript">
            var stiCore;
            function init(){
                stiCore = new STICore();
                stiCore.initialize();
            }
</script>
```

A reference in the body tag:

```
<body onload="init()">
```

And a set document nodes that should host parts the interface itself (please note that the following fragments need a / character before the last bracket to be valid XHTML):

For the map:

```
<div id="mapWindow" class="mapWindow">
```

For the Timeline

```
<div id="timeplot" class="timeplot">
```

For the table:

```
<div id="dataTables" class="dataTables">
```

HTTP

The HTTP based API is currently used only in a pull way, the includes XHR Proxy allows to use KML documents as described in M3.3.2.

It is envisioned to add a URL based API for the second prototype, this should allow referencing data and result sets as part of a browser request. This feature is needed to link to specific views and configurations of the interface.

JavaScript API

This documentation was generated from the sources. This is subject to change.

Class DataObject

Defined in DataObject.js

Constructor Summary

`DataObject(place, name, time, link, lon, lat)`
class that contains all needed information about an element of the resulting data set including time and place values

Method Summary

`setPercentage(percentage)`
sets the selection-percentage of the data object

Constructor Detail

DataObject

DataObject`(place, name, time, link, lon, lat)`
class that contains all needed information about an element of the resulting data set including time and place values

Parameters:

`String` `place` - the place, the data object corresponds to

`String` `name` - the name of the data object

`Date` `time` - the time stamp, the data object corresponds to

`String` `link` - an optional link to a data objects source

`float` `lon` - the longitude value corresponding to the given place

`float` `lat` - the latitude value corresponding to the given place

Method Detail

setPercentage

setPercentage(`percentage`)
sets the selection-percentage of the data object

Parameters:

`float` `percentage` - sets the percentage value (p); it describes the ratio to the actual selection
p = 0 if this data object is unselected, p = 1 if it is selected and 0 < p < 1 if its in a feather range

---

Class DataSet

Defined in DataSet.js

Constructor Summary

`DataSet`(`type, objects, notion`) class that represents all needed informations about a performed search

Method Summary

`addObject`(`object`)	adds an object to the data set; needed for individual data set
`changeHiddenStatus`()	changes the hidden state of this data set.

Constructor Detail

DataSet

DataSet(`type, objects, notion`)
class that represents all needed informations about a performed search

Parameters:

int `type` - is 0 for individual dataset and 1 for normal search

DataObject[] `objects` - corresponding data objects of this data set

String `notion` - is "Individual Data Set" or the typed search term

---

Method Detail

addObject

addObject(`object`)

adds an object to the data set; needed for individual data set

Parameters:

DataObject `object` - the object to add

---

changeHiddenStatus

changeHiddenStatus( )

changes the hidden state of this data set. if hidden value is true, the corresponding objects are not visible in map, timeplot and table

---

Class DisplayPointObject

Defined in PointObject.js

Constructor Summary

DisplayPointObject(`x, y, elements, radius, search`)

describes a finally displayed point object.

Method Summary

setOlPointFeature(`olPointFeature`)

sets the OpenLayers point feature for this point object to manage its selection status

setPointFeature(`pointFeature`)

sets the OpenLayers point feature for this point object

Constructor Detail

DisplayPointObject

DisplayPointObject(`x, y, elements, radius, search`)

describes a finally displayed point object. each instance of this class corresponds to a specific place and to only one data set

Parameters:

`float` `x` - the x (longitude) value of the point object

`float` `y` - the y (latitude) value of the point object

`DataObject[]` `elements` - array of data objects, that belong a point object instance

`int` `radius` - the resulting radius (in pixel) of the point in the map

`int` `search` - corresponding search index of the elements

Method Detail

setOlPointFeature

setOlPointFeature(`olPointFeature`)

sets the OpenLayers point feature for this point object to manage its selection status

Parameters:

`OpenLayers.Feature` `olPointFeature` - the overlay point feature for this object

---

setPointFeature

setPointFeature(`pointFeature`)

sets the OpenLayers point feature for this point object

Parameters:

`OpenLayers.Feature` `pointFeature` - the point feature for this object

Class ExtendedDataSource

Defined in ExtendedDataSource.js

Constructor Summary

`ExtendedDataSource``()`

data source to allow dynamic and manual creation of time slices, which will be used as input for the Simile Timeplot.

Method Summary

`initialize(dataSources, eventSources, objects, granularity)`
initializes the ExtendedDataSource

Constructor Detail

ExtendedDataSource

ExtendedDataSource( )
data source to allow dynamic and manual creation of time slices, which will be used as input for the Simile Timeplot. at the end we will get around 200 time slices, which units are depending on the time range of the input data

---

Method Detail

initialize

initialize`(dataSources, eventSources, objects, granularity)`
initializes the ExtendedDataSource

Parameters:

`Timeplot.ColumnSource[]` `dataSources` - the column sources corresponding to the data sets

`Timeplot.DefaultEventSource[]` `eventSources` - the event sources corresponding to the column sources

`Array` `objects` - the objects of all data sets

`SimileAjax.DateTime` `granularity` - the time granularity of the given data

Class PointObject

Defined in PointObject.js

Constructor Summary

PointObject() describes a temporal point object, that can save elements of different data sets for one place

Method Summary

addElement(searchIndex, element)	adds an elements with a specific searchIndex to this point object
addPoint(point)	adds another point object to this point object.
setRadius(radius)	sets the radius of this point object
setXY(x, y)	sets x (longitude) and y (latitude) values to this point object

Constructor Detail

PointObject

PointObject()

describes a temporal point object, that can save elements of different data sets for one place

---

Method Detail

addElement

addElement(searchIndex, element)

adds an elements with a specific searchIndex to this point object

Parameters:

int searchIndex - the search index of the actual dataset constellation

DataObject element - the data object to add

---

addPoint

addPoint(point)

adds another point object to this point object. here all elements of the other point object are inserted in the element array of this instance

Parameters:

PointObject `point` - another point object.

---

setRadius

setRadius`(radius)`

sets the radius of this point object

Parameters:

int `radius` - the radius to set

---

setXY

setXY`(x, y)`

sets x (longitude) and y (latitude) values to this point object

Parameters:

float `x` - the x (longitude) value

float `y` - the x (latitude) value

Class STICore

Defined in STICore.js

Constructor Summary

STICore`()`

defines the core component of the Spatio Temporal Interface

Method Summary

addElement`(object, granularity)`

adds an element to the user individual data set

changeVisibilityStatus`(index)`

changes the visibility status of a data set with specific index

deleteSearch(index) deletes a data set with specific index
initElements() initializes the sti components (map, timeplot, table) depending on the top masks of the data sets.
initialize() initializes the core component for the Spatio Temporal Interface.
parseKml(kmlFile) parses the kml-file which includes the results for a given search request
resetCore() resets the core within all elements and data objects to non-selection-status
setElementsTime(time) sets the time of an individual data object if it was selected through the timeplot element
updateTableAndMap() updates the table and map element.
updateTimeAndMap() updates the timeplot and map element.
updateTimeAndTable() updates the timeplot and table element.

Constructor Detail

STICore

STICore()

defines the core component of the Spatio Temporal Interface

Method Detail

addElement

addElement(object, granularity)

adds an element to the user individual data set

Parameters:

DataObject `object` - the data object to add

SimileAjax.DateTime `granularity` - the granularity of the data object's time value

---

changeVisibilityStatus

changeVisibilityStatus(`index`)

changes the visibility status of a data set with specific index

Parameters:

int `index` - the index of the data set, which visibility status should be changed

---

deleteSearch

deleteSearch(`index`)

deletes a data set with specific index

Parameters:

int `index` - the index of the data set to delete

---

initElements

initElements()

initializes the sti components (map, timeplot, table) depending on the top masks of the data sets. its called after a new search was performed, refining or undo button had been clicked

---

initialize

initialize()

initializes the core component for the Spatio Temporal Interface. here, the handling of the search interface is defined (including undo, refine and clear selection button). furthermore, the elements (map, timeplot, tables) are instanciated.

---

parseKml

parseKml(`kmlFile`)

parses the kml-file which includes the results for a given search request

Parameters:

File `kmlFile`

---

resetCore

resetCore()

resets the core within all elements and data objects to non-selection-status

---

setElementsTime

setElementsTime(`time`)

sets the time of an individual data object if it was selected through the timeplot element

Parameters:

`Date` `time` - the time value of the corresponding time slice the user clicked on

---

updateTableAndMap

updateTableAndMap()

updates the table and map element. its called from the STITimeplot object, when objects in the timeplot had been selected by timestamp or -range.

---

updateTimeAndMap

updateTimeAndMap()

updates the timeplot and map element. its called from the STITable object, when objects in one of the tables had been selected.

---

updateTimeAndTable

updateTimeAndTable()

updates the timeplot and table element. its called from the STIMap object, when objects on the map had been selected by featureSelect or polygon.

Class STIMap

Defined in STIMap.js

Constructor Summary

STIMap(`core`)

defines the map component of the Spatio Temporal Interface.

Method Summary

clearMap()

resets the map by destroying all elements

displayConnections(`leftTime, rightTime`)

should be used to display connections between data objects (is not working yet)

initialize() initializes the map for the Spatio Temporal Interface.
initObjectLayer(objects) initializes the object layer.
parseBaseLayers(xmlFile) parses all base layers in a given xmlFile
resetMap() resets the map by destroying all additional elements except the point objects, which are replaced
resetPoints() resets the point objects depending on the actual zoom level in basic style
setCanvas() sets the background canvas of the map window (or resets it after resizing the browser window)
updateByPlace(pointObjects, hover) updates the data objects percentages after a selection on the map had been performed
updateMap(zoom) updates the map, especially the object layer after selections had been executed in timeplot or table.

Constructor Detail

STIMap

STIMap(core)

defines the map component of the Spatio Temporal Interface. it builds a map context with the OpenLayers JavaScript Framework

Parameters:

STICore core - the sti core component, the map component has to deal with

---

Method Detail

clearMap

clearMap( )

resets the map by destroying all elements

---

displayConnections

displayConnections(`leftTime, rightTime`)

should be used to display connections between data objects (is not working yet)

Parameters:

`leftTime`

`rightTime`

---

initialize

initialize( )

initializes the map for the Spatio Temporal Interface. it includes setting up all layers of the map and defines all map specific interaction possibilities

---

initObjectLayer

initObjectLayer(`objects`)

initializes the object layer. here, all point representations for all zoom levels are calculated and initialized via OpenLayers.Feature

Parameters:

`objects`

---

parseBaseLayers

parseBaseLayers(`xmlFile`)

parses all base layers in a given xmlFile

Parameters:

`String` `xmlFile` - the name of the file to parse

---

resetMap

resetMap( )

resets the map by destroying all additional elements except the point objects, which are replaced

---

resetPoints

resetPoints( )

resets the point objects depending on the actual zoom level in basic style

---

setCanvas

setCanvas( )

sets the background canvas of the map window (or resets it after resizing the browser window)

---

updateByPlace

updateByPlace`(pointObjects, hover)`

updates the data objects percentages after a selection on the map had been performed

Parameters:

`pointObjects`

`hover`

---

updateMap

updateMap`(zoom)`

updates the map, especially the object layer after selections had been executed in timeplot or table. its called by the core component.

Parameters:

`zoom`

Class STITable

Defined in STITable.js

Constructor Summary

STITable`(core)` defines the table component of the Spatio Temporal Interface

Method Summary

deleteTables`()` deletes all tables
initialize`()` initializes the table component by creating a table with a single row, which cells will contain the tables of data sets
initTables`(objects, dataSets)`

	initializes the tables for given datasets
	`updateTables()` updates each dynamic table

Constructor Detail

STITable

STITable`(core)`

defines the table component of the Spatio Temporal Interface

Parameters:

`STICore` `core` - the sti core component, the table component has to deal with

---

Method Detail

deleteTables

deleteTables`()`

deletes all tables

---

initialize

initialize`()`

initializes the table component by creating a table with a single row, which cells will contain the tables of data sets

---

initTables

initTables`(objects, dataSets)`

initializes the tables for given datasets

Parameters:

`Array` `objects` - an array of object-array, which contain all elements that are in the actual display set

`DataSet[]` `dataSets` - all dataSets; needed for the header rows of the data tables

---

updateTables

updateTables`()`

updates each dynamic table

Class STITimeplot

Defined in STITimeplot.js

Constructor Summary

STITimeplot(core) defines the timeplot component of the Spatio Temporal Interface it builds a timeplot context with the Simile Widget Timeplot JavaScript Framework

Method Summary

addPlacePole(time) adds a place pole with specific time on the timeplot
hideTimeplot() hides the timeplot element (window) if all dataset had been deleted
initialize() initializes the timeplot for the Spatio Temporal Interface.
initTimeplot(objects, granularity) initializes the timeplot elements with arrays of objects, that have a specific time granularity
polesBySlices() updates the timeplot by setting place poles, after selections had been executed in map or table.
resetTimeplot() resets the timeplot to non selection status
setCanvas() sets the background canvas of the timeplot window (or resets it after resizing the browser window)
setFeather() calculates the new feather bounds

updateByTime()
updates the data objects percentages after a selection on the timeplot had been performed

Constructor Detail

STITimeplot

STITimeplot(core)
defines the timeplot component of the Spatio Temporal Interface it builds a timeplot context with the Simile Widget Timeplot JavaScript Framework

Parameters:

STICore core - the sti core component, the timeplot component has to deal with

---

Method Detail

addPlacePole

addPlacePole(time)
adds a place pole with specific time on the timeplot

Parameters:

Date time - the time the place pole should be placed is equivalent to one time slices time value

---

hideTimeplot

hideTimeplot()
hides the timeplot element (window) if all dataset had been deleted

---

initialize

initialize()
initializes the timeplot for the Spatio Temporal Interface. all elements (including their events) that are needed for user interaction are instanciated here, the slider element as well

---

initTimeplot

initTimeplot(objects, granularity)
initializes the timeplot elements with arrays of objects, that have a specific time granularity

Parameters:

Array objects - an array of object-array, which contain all elements that are in the actual display set

SimileAjax.DateTime granularity - the time granularity of the objects

---

polesBySlices

polesBySlices( )

updates the timeplot by setting place poles, after selections had been executed in map or table. its called by the core component.

---

resetTimeplot

resetTimeplot( )

resets the timeplot to non selection status

---

setCanvas

setCanvas( )

sets the background canvas of the timeplot window (or resets it after resizing the browser window)

---

setFeather

setFeather( )

calculates the new feather bounds

---

updateByTime

updateByTime( )

updates the data objects percentages after a selection on the timeplot had been performed

---

Class TimeSlice

Defined in ExtendedDataSource.js

Constructor Summary

TimeSlice(date)

small class that represents a time slice of the actual timeplot.

Constructor Detail

TimeSlice

TimeSlice(`date`)

small class that represents a time slice of the actual timeplot. it has a specific date and contains its corrsponding data objects as well

Parameters:

`Date` `date` - the date of the timeslice

HTTP API

The HTTP based approach for integrating third party data sets can be triggered via JavaScript and can work completely on the client side. But please be aware that, due to security consideration in the major browsers, a cross domain XML exchange isn't possible. To avoid this problem a so called XHR Proxy provided by KB will be part of the Europeana ThoughtLab implementation.

The HTTP interaction is implemented in the class STICore. Two methods can be used from JavaScript:

- `search()`
  This method recognieses HTTP URLs and calls `parseKML`() (se below) if needed. Otherwise the given string is passed to the configured search engine.
- parseKML()
  this method does the actual XML fetching and parsing.