SCHNITTSTELLENSPEZIFIKATION

Im Rahmen des Projekts „Interaktive Visualisierung von Open Data - IVOD“

31. DEZEMBER 2021

INSTITUT FÜR INTERNET-SICHERHEIT

Inhalt

Allgemeines

Das Dokument enthält neben der Spezifikation der eingehenden Datenstrukturen zur automatisierten Visualisierung, die Anleitung zum eigenen Betrieb und zur Nutzung der Schnittstelle, die im Rahmen des Projekts „Interaktive Visualisierung von Open Data – IVOD“, erstellt wurde. Die Schnittstelle wickelt den Prozess der automatisierten Erzeugung interaktiver Visualisierungen, sowie dem Vorschlagsystem zur Ermittlung geeigneter Visualisierungsmöglichkeiten bei bestimmten Eingabedaten ab. Die Art der Eingabedaten wird hier im Detail erläutert. Das Dokument richtet sich an Entwickler und Systemintegratoren, die einen fortlaufenden Betrieb der Schnittstelle sowie deren Weiterentwicklung unterstützen möchten. Die Informationen setzen daher im Großteil einen Kenntnistand in der Webentwicklung sowie der Nutzung von Serversystemen voraus. Im einleitenden Kapitel sind die Informationen zur Bereitstellung kompatibler Daten daher im allgemeinverständlichen Kontext erläutert.

Die Schnittstelle selbst kapselt die Funktionen der im Projekt erweiterten Open Source Software „pive“ (https://github.com/daboth/pive) und übernimmt die Bereitstellung der dort bestehenden Funktionen. Sie wurde zudem nach modernem Standard um IT-Sicherheitsfunktionalitäten ergänzt. Die Schnittstelle selbst wird unter der offenen BSD 2-Clause "Simplified" License veröffentlicht.

Der Quellcode der Schnittstelle wird nach Projektabschluss bereitgestellt als Open Source auf dem Portal Github unter der URL: https://github.com/internet-sicherheit/ivod-platform und kann im Anschluss an das Projekt von jedermann weiterentwickelt werden.

Lizenz

Datenstrukturen nach Visualisierungen

Dieser Abschnitt enthält Informationen über die derzeit verfügbaren Datenstrukturen, mit denen die Schnittstelle laut Spezifikation umgehen kann. Die hier erläuterten Datenstrukturen vermitteln eine Übersicht der implementierten Funktionalitäten der Schnittstelle, sowie notwendige Informationen zur zukünftigen Veröffentlichung offener Daten für die maschinelle Verarbeitung.

Die erweiterte Open Source Lösung pive bietet die automatische Zuordnung von geeigneten Visualisierungen auf den gegebenen Datensatz. Während des Vorgangs, werden alle Eigenschaften des Datensatzes auf Konfigurationen geprüft. Sobald alle Bedingungen der Konfigurationsdatei erfüllt sind, wird die Visualisierung in eine Ergebnisliste aufgenommen. Zusätzlich kann entschieden werden, ob ein Datensatz geeignet ist, um mehrere Reihen in einer einzelnen Visualisierung anzuzeigen, wie beispielsweise das Zeichnen mehrerer Verläufe in einem Liniendiagramm. Eine Kurzübersicht der Regelstruktur ist auch im Abschnitt „Formate“ vorhanden. Die hier aufgeführten Angaben zum Visualisierungsdatentyp entsprechen der Definition aus dem Abschnitt „Datentypen“. Die Angaben beziehen sich auf die Beschaffenheit eines einzelnen Datenpunktes in einer Datenreihe.

Barchart

Im Säulendiagramm werden Datenpunkte in einer ordinalen Skalierung durch Rechtecke angezeigt, deren Höhe mit dem Wert des Datenpunkts korreliert. Die Skalierung der Ordinatenachse beeinflusst dabei deren Höhe. Das Diagramm erwartet zur Generierung einen numerischen Parameter sowie eine zugehörige Zeichenkette. Datenpunkte unterscheiden sich anhand der Farbe ihrer Rechtecke. Es eignet sich gut zur Veranschaulichung relativer und absoluter Häufigkeiten.

Datenfeld	Visualisierungsdatentyp
Wert	Zahl
Bezeichnung	Text

Bubblechart

Das Blasendiagramm zeigt Datenpunkte bestehend aus mindestens drei numerischen Parametern und besitzt zwei skalierbare Achsen. Datumsangaben werden von der Abszisse akzeptiert. Datenpunkte bestehen aus Abszisse, Ordinate und einem Radius des Kreises, der durch die dritte Variable definiert wird. Für eine bessere Erkennbarkeit der Punkte, lässt sich die Opazität der Kreise einstellen, Datenpunkte die vollständig von anderen verdeckt werden, sind so noch immer sichtbar. Entsprechen die Parameter Vielfachen der geforderten Variablenanzahl, abzüglich ihrer gemeinsamen Abszisse, werden sie als zusätzliche Datenreihe in das Diagramm mit andersfarbigen Kreisen aufgenommen.

Datenfeld	Visualisierungsdatentyp
X	Zahl
Y1	Zahl
Radius1	Zahl
Y2	Zahl
Radius2	Zahl
…	…
Y_n	Zahl
Radius_n	Zahl

Chordchart

Das Saitendiagramm veranschaulicht die Beziehungen von Gruppen aus Entitäten. Es erwartet zwei Zeichenketten und einen numerischen Parameterzur Darstellung. Die eingehende Datenstruktur bezeichnet dabei einen gewichteten Graphen.

Die Verbindungen des Graphen werden dann in einem zirkulären Diagramm angezeigt, wobei gleiche Knoten gruppiert werden. Die Gruppen sind im äußeren Ring des Diagramms angeordnet, während deren Verbindungen untereinander sich durch feine Linien (Saiten) zeigen, deren anfängliche Dicke abhängig von der Gewichtung ist. Dabei werden alle ein- und ausgehenden Verbindungen, sowie die eigenen Verbindungen innerhalb der Gruppe, berücksichtigt.

Datenfeld	Visualisierungsdatentyp
Start	Text
Ziel	Text
Gewichtung	Zahl

Heatmap

Die Heatmap zeichnet Polygone anhand der Auflösung von Gebietsnamen auf entsprechende GeoJSON Koordinaten und färbt diese entsprechend ihrer Intensität ein. Zum Betrieb der Heatmap wird ein eigener Geodatenserver wie Overpass (https://wiki.openstreetmap.org/wiki/Overpass_API) notwendig, um uneingeschränkt auf die Open Streetmap Daten zur Namensauflösung zurückzugreifen. Sie zeichnet sich durch die farbliche Kennzeichnung von angrenzenden Gebieten aus und stellt eine grafische Repräsentation von Vergleichen, der den Gebieten zugeordneten Zahlenwerten, dar. Die Angabe der verschiedenen Gebietsnamen und mindestens jeweils eines zugehörigen Zahlenwerts pro Datenpunkt ist notwendig, um diesen Vergleich vorzunehmen. Sind mehrere numerische Werte angegeben, wird der jeweils erste als Vergleichswert zur Erzeugung der Heatmap bezogen.

Datenfeld	Visualisierungsdatentyp
Name1	Text
Wert1	Zahl
Wert2	Zahl
…	…
Wert_n	Zahl

Linechart

Einzelne Datenpunkte der Datenreihe bestehen je aus Abszisse und Ordinate und werden durch Linien miteinander verbunden. Es sind mindestens zwei numerische Parameter pro Datenpunkt notwendig, jeder weitere Zahlenwert repräsentiert eine neue Datenreihe. Optional akzeptiert die Abszisse auch Datumsangaben. Bei gleichzeitiger Darstellung mehrerer Datenreihen, werden sie in Form verschiedener Linien dargestellt. Ordinaten repräsentieren die unterschiedlichen Daten an diesem Punkt. Die Achsen beide lassen sich unterschiedlich skalieren.

Datenfeld	Visualisierungsdatentyp
X	Zahl
Y1	Zahl
Y2	Zahl
…	…
Y_n	Zahl

Piechart

Das Kreis-, oder auch Tortendiagramm zeigt Datenpunkte als Teil eines vollständigen Kreises. Es benötigt zwei Parameter, einen numerischen und einen Text. Die Kreisfläche selbst zeigt 100% der gesamten numerischen Daten, während die Kreissegmentflächen aus den prozentualen Anteilen der einzelnen Datenpunkten bestehen. Datenpunkte unterscheiden sich dabei durch ihre Kolorierung. Es eignet sich daher gut zur Darstellung relativer Häufigkeiten.

Datenfeld	Visualisierungsdatentyp
Wert	Zahl
Bezeichnung	Text

Point-of-Interest (POI)

Die Visualisierungsform POI bildet Standorte auf einer Karte ab. Die Standorte werden durch Koordinatenpaare beschrieben und auf einer Landkarte dargestellt. Wie auch bei der Heatmap wird durch die Namensauflösung der GeoJSON Informationen der Betrieb eines Overpass Servers notwendig. Voraussetzung zum Zeichnen ist der Beginn mit dem jeweiligen Breitengrad, gefolgt vom Längengrad des Datenpunktes. Optional können, daran anknüpfend, die den Standort beschreibenden Felder als Zeichenkette abgelegt werden.

Datenfeld	Visualisierungsdatentyp
Latitude	Breitengrad
Longitude	Längengrad
Bezeichner1	Text
Bezeichner2	Text
…	…
Bezeichner_n	Text

Polygon

Polygondiagramme identifizieren Bereiche auf einer Landkarte. So lassen sich etwa bestimmte Bereiche einer Landkarte gezielt hervorheben. Die Landkarte bildet sich aus einer Menge an Polygonen, die jeweils ein Gebiet markieren. Bezeichner und weitere Eigenschaften der Datenpunkte sind im GeoJSON Objekt (Polygon) selbst gekapselt.

Datenfeld	Visualisierungsdatentyp
Wert1	Polygon
Wert2	Polygon
…	…
Wert_n	Polygon

Scatterchart

Das Streudiagramm besitzt zwei unterschiedlich skalierbare Achsen und benötigt pro Datenpunkt mindestens zwei numerische Parameter. Alternativ werden von der Abszisse auch Datumsangaben akzeptiert. Einzelne Datenpunkte werden als Punkt in das Diagramm gezeichnet und bestehen aus einer beliebig geordneten Abszisse und deren Ordinate. Werden pro Abszisse mehrere Ordinaten angegeben, werden sie als weiter Datensätze in Form andersfarbiger Punkte im Diagramm abgebildet. Der Radius der Punkte kann selbst angegeben werden.
Streudiagramme werden für verteilte Daten benutzt, die keiner direkten Ordnung folgen. So lassen sich die Werte zweier Variablen vergleichen, oder auch in Kombination mit anderen Daten betrachten. Zusammenhänge und Merkmale der Daten sind in einem solchen Diagramm erkennbar. Verteilen sich Punkte sichtbar auf distinktive Bereiche des Diagramms, spricht man beispielsweise von einer Gruppierung der Daten.

Datenfeld	Visualisierungsdatentyp
X	Zahl
Y1	Zahl
Y2	Zahl
…	…
Y_n	Zahl

Benutzung der Schnittstelle

Für die generelle Nutzung der Schnittstelle (API) wird ein Benutzersystem bereitgestellt, dass durch moderne Authentifizierungsmechanismen gesichert ist. Dies erfordert sowohl Management der API-Tokens, wie auch die Verwaltung von Benutzern, um Zugriff auf die Mechanismen zu erlangen. Der Prozess wird in den folgenden Abschnitten im Detail erläutert. Zu Projektabschluss steht die Schnittstelle bereits betriebsbereit unter https://visquid.org zur Verfügung. Wird die Schnittstelle selbst auf einem Server aufgesetzt, ist die URL der Endpunkte an den neuen Host anzupassen.

Authentifizierung

Für die Nutzung der hauptsächlichen Funktionen ist eine Authentifizierung des Benutzer-Clients erforderlich. Bei Anfragen, die an die Schnittstelle gestellt werden, ist ein Java Web Token Cookie zu setzen und zu übergeben. Für den Erhalt des Tokens, ist die Sendung eines Request an folgenden Endpunkt notwendig:

URL: https://visquid.org/api/token/
Method: POST
Content-Type: application/json
Body:

```
{
    email: <USER_EMAIL>,
    password: <USER_PASSWORD>
}
```

Ist die Authentifizierung erfolgreich, befindet sich im Response Body der Nachricht ein JSON-Objekt. Das Token befindet sich unter dem Schlüssel "token". Jedes Token erhält eine bestimmte Lebensdauer, sofern der Benutzer-Client dies unterstützt, wird das Cookie zusätzlich als „Secure HttpOnly-Cookie“ mit der Lebensdauer des Tokens gesetzt. So wird sichergestellt, dass ein Cookie im Browsercontext weder ausgelesen noch verändert werden kann.

Token Refresh

Die zur Verfügung gestellten Sicherheitstoken enthalten je eine eigene Lebensdauer. Das Token besitzt eine Lebensdauer von einer Stunde, kann jedoch bis zu sieben Tage lang erneuert werden. Dazu muss ein Request an folgenden Endpunkt gesendet werden:

URL: https://visquid.org/api/token/refresh/
Method: POST
Content-Type: application/json

Logout/Token Invalidation

Während der Nutzung der Schnittstelle ist keine aktive Session im Browser vorhanden, somit ist ein Logout im klassischen Sinne nicht nötig oder möglich. Es ist daher ausreichend, das Token nicht mehr weiter mitzusenden. Um das Token jedoch vorzeitig zu invalidieren, ist die Nutzung eines Blacklist Verfahrens möglich. Dazu ist ein Request an folgenden Endpunkt notwendig:

URL: https://visquid.org/api/token/blacklist/
Method: POST
Content-Type: application/json

So der Nutzerclient dies zulässt, wird die Cookie-Lebenszeit nun auf 0 Sekunden gesetzt und es gilt fortan als ungültig.

Visualisierung

Zur Erzeugung einer Visualisierung mit bestehenden Daten sind derzeit vier Arbeitsschritte vorgesehen, die in dieser Reihenfolge ablaufen:

- Hochladen der Daten
- Auflistung der möglichen Visualisierungen empfangen
- (Optional) Änderung der Konfiguration der Visualisierung
- Erzeugung der gewünschten Visualisierung mit optionaler Konfiguration und Empfang des erstellen Programmcodes (Java-Script)
- Einbindung der Visualisierung in ein bestehendes Webdokument durch Integration des Programmcodes.

Hochladen der Daten

Stehen die Daten bereit, können sie an einen Schnittstellen Endpunkt übermittelt werden. Dazu ist die Authentifizierung eines Nutzers erforderlich, damit die notwendigen Rechte zur Verarbeitung der Daten gesetzt sind. Der Endpunkt ist im Folgenden beschrieben:

URL: https://visquid.org/api/datasources
Method: POST
Content-Type: application/json
Body:

```
{
    datasource_name: <NAME>,
    data: <DATA>
}
```

Das gekapselte Visualisierungswerkzeug pive liest Daten in den Formaten JSON und CSV ein. Dabei wird ein für den Nutzer eindeutiger String übermittelt. Die werden als BASE64-kodierter String eingelesen, sie müssen in diesem Format vorliegen. Bei erfolgreicher Erstellung der Daten, enthält der Response Body der

Serverantwort ein Datasource JSON-Objekt. Mehr Details dazu finden sich unter der API-Spezifikation in den folgenden Abschnitten.

Auflistung der möglichen Visualisierungen

Enthält der Nutzer Zugriffsrechte auf die Datasource kann er sich auflisten lassen, welche Visualisierungen derzeit für diese Daten möglich sind. Derzeit ist vorgesehen, dass jeder Nutzer automatisch Zugriff auf die von ihm selbst hochgeladenen Daten erhält. Hierzu ist die Sendung eines Requests an den folgenden Endpunkt nötig:

URL: https://visquid.org/api/datasources/<DATASOURCE_ID>/charttypes
Method: GET
Content-Type: application/json

Dabei ist <DATASOURCE_ID> die ID der Datasource. Bei erfolgreichem Call enthält die Antwort eine Liste an lesbaren Strings mit Namen von möglichen Visualisierungen.

Erzeugung der gewünschten Visualisierung mit optionaler Konfiguration

Die Konfiguration der Visualisierung erfolgt optional. Die dazu notwendigen Einstellmöglichkeiten, wie etwa die Anpassung von Farben, werden in einem späteren Abschnitt behandelt.

Jede Visualisierung wird mit einer von vier möglichen Sichtbarkeitsstufen erzeugt:

- 0 – Privat, die Visualisierung steht nur dem eigenen Benutzer zur Verfügung
- 1 – Explizit geteilt. Die Visualisierung kann durch weitere Benutzer abgerufen werden, mit denen diese Visualisierung explizit geteilt wurde.
- 2 – Explizit geteilt und abrufbar über einen Direktlink.
- 3 – Öffentlich. Die Visualisierung ist öffentlich einsehbar.

Durch die geeigneten Zugriffsrechte auf die Datasource, kann eine Visualisierung dann erstellt werden. Dazu wird ein Request an folgenden Endpunkt gesendet:

URL: https://visquid.org/api/datasources
Method: POST
Content-Type: application/json
Body:

```
{
    datasource: <DATASOURCE_ID>,
    chart_type: <CHART_TYPE>,
    config: <CHART_CONFIG>,
    chart_name: <CHART_NAME>,
    downloadable: <DOWNLOADABLE>,
    visibility: <VISIBILITY>
}
```

Felderbeschreibung:

- <DATASOURCE_ID> die ID der Datasource,
- <CHART_TYPE> einer der Visualisierungsnamen aus dem vorherigen Schritt
- <CHART_CONFIG> ein JSON-Objekt, dass die Parameter für die Visualisierung enthalten. Diese Konfiguration kann angepasst werden (Optionaler Schritt).

- <CHART_NAME> eine für den Nutzer eindeutige Benennung für die Visualisierung, um sie später abzurufen.
- <DOWNLOADABLE> ein Boolean, der angibt, ob die Originaldaten im Rohformat heruntergeladen werden können.
- <VISIBILITY> die allgemeine Sichtbarkeit als Integer

Bei erfolgreicher Erstellung enthält der Response Body ein Chart JSON-Objekt. Mehr Details hierzu finden sich in einem späteren Abschnitt der API-Spezifikation.

Einbindung der Visualisierung

Mit Zugriffsrechten auf die Visualisierung kann diese nun in ein Webdokument eingebunden werden. Da die Entwicklung im Ursprung auf der Java-Script Bibliothek Data Driven Documents D3.js (https://d3js.org/) basiert, muss diese Technologie ebenfalls im Zieldokument vorhanden sein.

Generell werden die folgende Abhängigkeiten im Dokument benötigt.

- https://visquid.org/api/code/base.js
- https://d3js.org/d3.v6.min.js
- https://d3js.org/d3.hive.v0.min.js

Zusätzlich muss die für die speziell ausgewählte Visualisierung benötigte Abhängigkeit geladen werden. Hierzu kann die URL https://visquid.org/api/charts/<CHART_ID>/code aufgelöst werden, wobei <CHART_ID> der ID der Visualisierung entspricht. Dieser Aufruf leitet auf die entsprechend benötigte Java-Script Datei für diese Visualisierung weiter.

Anschließend wird die Konfiguration der Visualisierung benötigt. Diese ist als JSON-Objekt verfügbar unter https://visquid.org/api/charts/<CHART_ID>/config

Die Konfiguration selbst ist optional anpassbar und enthält Felder für die Informationen über verwendete Farben, Größenangaben der gewünschten Erzeugung sowie weitere Parameter die in späterem Abschnitt genauer beschrieben werden.

Zuletzt kann das Visualisierungsobjekt als Variable im Zieldokument angelegt werden. Die Schnittstelle steuert die Erzeugung und Bereitstellung des notwendigen Quellcodes zur Anzeige. Dazu wird der folgende Aufruf im Quelltext des Zieldokuments nötig:

```
let chart = window.pive.createChartFromConfig(config);
```

Die gewünschte Visualisierung ist nun im Quelltext vorhanden und kann im Anschluss mit dem folgenden Aufruf angezeigt werden:

```
chart.render();
```

Die Auflösung von Abhängigkeiten und Konfiguration kann entweder einmalig serverseitig durchgeführt werden und beispielsweise durch Templating in die entsprechenden Webdokumente eingebaut werden, oder dynamisch bei jedem Seitenaufruf durchgeführt werden.

Endpunkte

Dieser Abschnitt dokumentiert die Endpunkte der Schnittstelle und führt die möglichen Parameter auf, die bei einem Request angefragt und in der Serverantwort übermittelt werden. Die Beschreibung der Endpunkt erfolgt für den internationalen Gebrauch im üblichen Standard einer Endpunktbeschreibung in Englisch, da der Quellcode im Zuge der Veröffentlichung als Open Source zugänglich gemacht wird. Die Statuscode Nummerierung erfolgt ebenfalls im Standard. Zur Förderung der Lesbarkeit wird jedoch eine kurze Beschreibung der Punkte auf Deutsch benannt.

chart-add

Der Endpunkt ermöglicht eine neue Visualisierung hinzuzufügen oder eine Liste von bisherigen erstellten Visualisierungen anzufordern.

url: charts

Description: List charts or add a new chart

methods: [GET, POST]

GET:

Parameters:

'chart_type':

Type: query

Description: Filter for charts with this exact name

'modification_time__lte':

Type: query

Description: Filter for charts last modified before the passed timestamp argument

'modification_time__gte':

Type: query

Description: Filter for charts last modified after the passed timestamp argument

'creation_time_lte':

Type: query

Description: Filter for charts created before the passed timestamp argument

'creation_time__gte':

Type: query

Description: Filter for charts created after the passed timestamp argument

Returns:

Format: JSON

Type: [Chart]

Code: 200

POST:

Parameters:

'datasource':

Description: ID of datasource to be used for this chart

Type: int

'chart_type':

Description:Name of the chart type to be used, see pive for list of allowed values

Type: string

'config':

Description: Parameters for chart rendering

Type: JSON Object

'chart_name':

Description: Object name for displaying/ordering elements in UI

Type: string

'downloadable':

Description: Determines if the (processed) data can be downloaded

Type: boolean

Default: False

'visibility':

Description: Determines the share level of this chart

Type: Enum(Private, Shared, Semi-Public, Public)

Default: 0

Returns:

Format: JSON

Type: Chart

Code: 201

chart-get

Mit diesem Endpunkt kann eine bestehende Visualisierung angezeigt, geändert oder gelöscht werden.

url: charts/<IsD>

Description: Show, alter or delete a specific chart

methods: [GET, PATCH, DELETE]

GET:

Returns:

Format: JSON

Type: Chart

Code: 200

PATCH:

Parameters:

'config':

Description: Parameters for chart rendering

Type: JSON Object

Default: Prior Value

‘chart_name':

Description: Object name for displaying/ordering elements in UI
Type: string

Default: Prior Value

'downloadable':

Description: Determines if the (processed) data can be downloaded

Type: boolean

Default: Prior Value

'visibility':

Description: Determines the share level of this chart

Type: Enum(Private, Shared, Semi-Public, Public)

Default: Prior Value

Returns:

Format: JSON

Type: Chart

Code: 200

DELETE:

Returns:

Code: 204

chart-shared

Anzeigen, mit wem eine bestimmte Visualisierung bisher geteilt wurde. Ermöglicht zudem, weitere Nutzer zur Teilung hinzuzufügen oder zu entfernen. Es ist möglich, Nutzergruppen anzulegen, für die eine Visualisierung sichtbar ist.

url: charts/<ID/shared/>

Description: Show, add, or remove shares from a chart

methods: [GET, PATCH, DELETE]

GET:

Returns:

Format: JSON

Type: Shares

Code: 200

PATCH:

Parameters:

'users':

Description: List of user ids to add to the share

Type: [uuid]

Default: []

'groups':

Description: List of group ids to add to the share

Type: [int]

Default: []

Returns:

Format: JSON

Type: Shares

Code: 200

DELETE

Parameters:

'users':

Description: List of user ids to remove from the share

Type: [uuid]

Default: []

'groups':

Description: List of group ids to remove from the share

Type: [int]

Default: []

Returns:

Format: JSON

Type: Shares

Code: 200

chart-data

Liefert die Daten, die für die Visualisierung durch die Schnittstelle aufbereitet wurden im JSON Format.

url: charts/<ID>/data

Description: Get processed data for displaying

methods: [GET]

GET:

Returns:

Format: JSON

Type: String

Code: 200

chart-code

Ermöglicht Zugriff auf den rohen Java-Script Code, der dieser Visualisierung zugeordnet ist.

url: charts/<ID>/code

Description: Redirects to the javascript code associated with this chart.
See **code-get**

methods: [GET]

GET:

Returns:

Code: 302

chart-config

Liefert die aktuelle Konfiguration der Visualisierung als JSON.

url: charts/<ID>/config

Description: Get config file for this chart

methods: [GET]

GET:

Returns:

Format: JSON

Type: String

Code: 200

chart-files

Liefert eine automatisiert erstellte Datei, die mit der Visualisierung verknüpft ist.

url: charts/<ID>/files/

Description: Get a file associated with this chart.
This allows pive visualisations to add more files. Uses a whitelist for filenames.

methods: [GET]

GET:

Returns:

Format: JSON

Type: octet-stream

Code: 200

code-common-get

Liefert gemeinsame Code Dateien für alle Visualisierungen.

url: code/<name>

Description: Returns code files common to all charts.

methods: [GET]

GET:

Returns:

Format: Javascript

Type: octet-stream

Code: 200

code-get

Liefert den reinen Java-Script Code zur Darstellung eines Diagramms.

url: code/<version>/<name>

Description: Returns the javascript to visualise a chart. **chart-code** redirects here. Normally this does not need to be called manually

methods: [GET]

GET:

Returns:

Format: Javascript

Type: octet-stream

Code: 200

datasource-add

Liefert eine Liste von Datenquellen, oder fügt eine neue hinzu.

url: datasources

Description: List datasources or add a new datasource

methods: [GET, POST]

GET:

Returns:

Format: JSON

Type: [Datasource]

Code: 200

POST:

Parameters:

'datasource_name':

Description: Object name for displaying/ordering elements in UI

Type: string

OneOf:

'url':

Description: URL that points to the datasource

Type: URL

'data':

Description: Data to be used as datasource, base64 encoded

Type: string

'visibility':

Description: Determines the share level of this datasource

Type: Enum(Private, Shared, Semi-Public, Public)

Default: 0

Returns:

Format: JSON

Type: Datasource

Code: 201

datasource-get

Anzeigen oder Löschen einer Datenquelle.

url: datasources/<ID>

Description: Show or delete a specific datasource

methods: [GET, DELETE]

GET:

Returns:

Format: JSON

Type: Datasource

Code: 200

PATCH:

Parameters:

'datasource_name':

Description: Object name for displaying/ordering elements in UI

Type: string

Default: Prior Value

'visibility':

Description: Determines the share level of this datasource

Type: Enum(Private, Shared, Semi-Public, Public)

Default: Prior Value

Returns:

Format: JSON

Type: Datasource

Code: 200

DELETE:

Returns:

Code: 204

datasource-shared

Anzeigen, Hinzufügen oder Entfernen von Nutzern, mit denen diese Datenquelle geteilt wird.

url: datasources/<ID>/shared

Description: Show, add or remove users this datasource is shared with

methods: [GET, PATCH, DELETE]

GET:

Returns:

Format: JSON

Type: Shares

Code: 200

PATCH:

Parameters:

'users':

Description: List of user IDs to add to share

Type: [int]

Default: []

'groups':

Description: List of group IDs to add to share

Type: [int]

Default: []

Returns:

Format: JSON

Type: Shares

Code: 200

DELETE:

Parameters:

'users':

Description: List of user IDs to remove from share

Type: [int]

Default: []

'groups':

Description: List of group IDs to remove from share

Type: [int]

Default: []

Returns:

Format: JSON

Type: Shares

Code: 200

datasource-charttypes

Anzeigen der möglichen Visualisierungsmöglichkeiten für eine Datenquelle

url: datasources/<ID>/charttypes

Description: Show the chart types this datasource can be visualised with

methods: [GET]

GET:

Returns:

Format: JSON

Type: [String]

Code: 200

dashboard-add

Dashboards auflisten oder ein neues Erstellen.

url: dashboard

Description: List dashboards or add a new dashboard

methods: [GET, POST]

GET:

Returns:

Format: JSON

Type: [Dashboard]

Code: 200

Parameters:

'config':

Description: Dashboard config

Type: JSON Object

'name':

Description: Object for displaying/ordering elements in UI

Type: string

'visibility':

Description: Determines the share level of this dashboard

Type: Enum(Private, Shared, Semi-Public, Public)

Default: 0

Returns:

Format: JSON

Type: Dashboard

Code: 201

dashboard-get

Anzeigen oder löschen eines bestimmten Dashboards

url: dashboard/<ID>

Description: Show or delete a specific dashboard

methods: [GET, DELETE]

GET:

Returns:

Format: JSON

Type: Datasource

Code: 200

PATCH:

Parameters:

'config':

Description: Dashboard config

Type: JSON Object

Default: Prior Value

'name':

Description: Object name for displaying/ordering elements in UI

Type: string

Default: Prior Value

'visibility':

Description: Determines the share level of this dashboard

Type: Enum(Private, Shared, Semi-Public, Public)

Default: Prior Value

Returns:

Format: JSON

Type: Datasource

Code: 200

DELETE:

Returns:

Code: 204

dashboard-shared

Anzeigen, Hinzufügen oder Entfernen von Nutzern mit denen das Dashboard geteilt wird.

url: dashboard/<ID>/shared

Description: Show, add or remove users this dashboard is shared with

methods: [GET, PATCH, DELETE]

GET:

Returns:

Format: JSON

Type: Shares

Code: 200

PATCH:

Parameters:

'users':

Description: List of user IDs to add to share

Type: [int]

Default: []

'groups':

Description: List of group IDs to add to share

Type: [int]

Default: []

Returns:

Format: JSON

Type: Shares

Code: 200

DELETE:

Parameters:

'users':

Description: List of user IDs to remove from share

Type: [int]

Default: []

'groups':

Description: List of group IDs to remove from share

Type: [int]

Default: []

Returns:

Format: JSON

Type: Shares

Code: 200

sharegroup-add

Gruppenshares einsehen oder neue erstellen.

url: groups

Description: List sharegroups available to you or create a new one

methods: [GET, POST]

GET:

Returns:

Format: JSON

Type: [ShareGroup]

Code: 200

POST:

Parameters:

'name':

Description: Name of new sharegroup

Type: string

'is_public':

Description: Privacy setting of new sharegroup

Type: boolean

Default: false

'group_admins':

Description: List of user IDs of users with privileged group access

Type: [uuid]

Default: []

'group_members':

Description: List of user IDs of users with group access

Type: [uuid]

Default: []

Returns:

Format: JSON

Type: ShareGroup

Code: 200

sharegroup-get

Erhalten, Updaten oder Löschen einer Sharegroup

url: groups/<pk>

Description: Retrieve, update or delete a sharegroup

methods: [GET, PATCH, DELETE]

GET:

Returns:

Format: JSON

Type: [ShareGroup]

Code: 200

POST:

Parameters:

'is_public':

Description: Privacy setting of new sharegroup

Type: boolean

Default: false

Returns:

Format: JSON

Type: ShareGroup

Code: 200

POST:

Returns:

Code: 204

sharegroup-properties

Erhalten oder ändern von Sharegroup Mitgliedern. Entsprechende Nutzungsrechte der Gruppe vorrausgesetzt.

url: groups/<pk>/properties

Description: Retrieve or alter sharegroup members, requires elevated permissions on the group

methods: [GET, PATCH, DELETE]

GET:

Returns:

Format: JSON

Type: ShareGroup

Code: 200

PATCH:

Parameters:

'group_admins':

Description: List of user IDs of users with privileged group access

Type: [uuid]

Default: []

'group_members': - Description: List of user IDs of users with group access - Type: [uuid] - Default: []

Returns:

Format: JSON

Type: ShareGroup

Code: 200

DELETE:

Parameters: - 'group_admins': - Description: List of user IDs of users with privileged group access - Type: [uuid] - Default: [] - 'group_members': - Description: List of user IDs of users with group access - Type: [uuid] - Default: []

Returns:

Format: JSON

Type: ShareGroup

Code: 200

token_obtain

Unter Django Rest Framework sind die Tokens dokumentiert (https://django-rest-framework-simplejwt.readthedocs.io/en/latest/). Dieser Endpunkt dient als Login.

url: token/

Description: See drf-jwt documentation for obtain_jwt_token. Serves as login endpoint

token_refresh

Siehe token_obtain. Dieser Endpunkt ermöglicht das kopieren von Token.

url: token/refresh/

Description: See drf-jwt documentation for refresh_jwt_token. Token will be copied from cookie to request object by middleware

token_verify

Siehe token_obtain. Dient dem Verifizieren von Token.

url: token/verify/

Description: See drf-jwt documentation for verify_jwt_token. Token will be verified.

token_blacklist

Siehe token_obtain. Dient dem Blacklisting von Tokens und dient als Logout.

url: token/blacklist/

Description: See drf-jwt documentation for BlacklistView. Serves as logout endpoint. Middleware will tell client to remove token.

get_current_user

Den aktuellen Benutzer erhalten und unprivilegierte Informationen updaten.

url: user/me/

Description: Get the current user and update unprivileged information

methods: [GET, PATCH]

GET:

Returns:

Format: JSON

Type: User

Code: 200

PATCH:

Parameters:

'username':

Description: Display name

Type: string

Default: Previous value

'first_name':

Description: First name

Type: string

Default: Previous value

'last_name':

Description: Last name

Type: string

Default: Previous value

'real_name':

Description: Flag indicating if first_name and last_name should generally be served on lookup and used for user search

Type: boolean

Default: Previous value

'public_profile':

Description: Flag indicating if profile should turn up in search. Private profiles are still searchable by full id

Type: boolean

Default: Previous value

Returns:

Format: JSON

Type: User

Code: 200

change_password

Endpunkt, um das Passwort des aktuellen Nutzers zu ändern.

url: user/me/password/

Description: Change the password of the current user

methods: [POST]

POST:

Parameters:

'oldPassword':

Description: Former password, used to reauthenticate the user for this sensitive request

Type: string

'newPassword':

Description: New password

Type: string

Returns:

Code: 200

change_email

Ändern der Nutzeremail.

url: user/me/email/

Description: Send an e-mail change request for the current user

methods: [POST]

POST:

Parameters:

'password':

Description: Current password, used to reauthenticate the user for this sensitive request

Type: string

'newEmail':

Description: New e-mail. This mail will receive a verification e-mail. -
Type: e-mail

Returns:

Code: 200

get_user

Nutzer anhand seiner ID erhalten.

url: user/id/<pk>

Description: Retrieve a user object by its id

methods: [GET]

GET:

Returns:

Format: JSON

Type: User

Code: 200

search_user_by_name

Nutzer anhand seines Namens erhalten.

url: user/search/

Description: Retrieve a user object by its id

methods: [GET]

GET:

Returns:

Format: JSON

Type: User

Code: 200

get_users

Mehrer Nutzer gleichzeitig erfassen anhand eines Arrays von uuid.

url: user/

Description: Get multiple user objects for an array of uuid

methods: [POST]

POST:

Parameters:

'users':

Description: List of user uuids

Type: [uuid]

Returns:

Format: JSON

Type: [User]

Code: 200

iniate_password_reset

Passwort Wiederherstellung iniitieren.

url: password/reset/

Description: Try to create a password reset request. This request will return 200 even if the provided mail is not associated with any accounts

methods: [POST]

POST:

Parameters:

'email':

Description: E-Mail address of the account in question

Type: [uuid]

Returns:

Code: 200

do_password_reset

Passwort Wiederherstellung durchführen.

url: password/reset/<reset_id>/

Description: Change the password as recovery. The GET-Endpoint is not an API-Endpoint, but a minimal password change page

methods: [GET, POST]

GET:

Returns:

Code: 200

POST:

Parameters:

'password':

Description: The new password to set for the account associated with this reset

Type: string

Returns:

Code: 200

confirm_email

Email bestätigen.

url: email/confirm/<token>/

Description: Confirm the password change request, verifying the user has access to the new e-mail address

methods: [GET]

GET:

Returns:

Code: 200

Objektklassen

In der Schnittstelle sind eigene Objektklassen angelegt worden, um die Verarbeitung und Zusammenführung auf Nutzerebene, Datensammlungen, Shares und Dashboards zu ermöglichen. Es werden die folgenden Datentypen verarbeitet. Ihre Verwendung entspricht ihrer Definition der Eigenschaften in den folgenden Abschnitten.

- Datasource – Beschreibt eine Datenquelle.
- Chart – Beschreibt eine Visualisierung inklusive ihrer Komponenten.
- Shares – Geteilte Inhalte und die verknüpften Privilegien.
- Dashboard – Zusammenführungen mehrerer Visualisierungen.

Datasource

'id':

Description: Database id of datasource

Type: int

'source':

Description: (Original) Source of data

Type: URL

'datasource_name':

Description: Object name for displaying/ordering elements in UI

Type: string

'owner'

Description: Database id of owner

Type: uuid

'creation_time'

Description: Timestamp of when this datasource was created

Type: string

'modification_time'

Description: Timestamp of when this datasource was last modified

Type: string

Chart

'id':

Description: Database id of chart

Type: int

'chart_type':

Description: Name of the chart type to be used, see pive for list of allowed values

Type: string

'chart_name':

Description: Object name for displaying/ordering elements in UI

Type: string

'owner'

Description: Database id of owner

Type: uuid

'original_datasource':

Description: Database id of the datasource the chart is based on, can be null, if datasource has been removed

Type: int

'config':

Description: Parameters for chart rendering

Type: JSON Object

'downloadable':

Description: Determines if the (processed) data can be downloaded

Type: boolean

Default: False

'visibility':

Description: Determines the share level of this chart

Type: Enum(Private, Shared, Semi-Public, Public)

Default: 0

'creation_time'

Description: Timestamp of when this chart was created

Type: string

'modification_time'

Description: Timestamp of when this chart was last modified

Type: string

ShareGroup

'id':

Description: Group ID

Type: int

'owner':

Description: ID of the group creator, can currently not be changed afterwards

Type: uuid

'name':

Description: Display name of this group

Type: string

'group_admins':

Description: List of user IDs of users with privileged group access

Type: [uuid]

'group_members':

Description: List of user IDs of users with group access

Type: [uuid]

'is_public':

Description: List of user IDs of users with group access

Type: boolean

Default: false

Shares

'users':

Description: List of user IDs this share applies to

Type: [uuid]

Default: []

'groups':

Description: List of group IDs this share applies to

Type: [int]

Default: []

Dashboard

'id':

Description: Database id of dashboard

Type: uui

'name':

Description: Object name for displaying/ordering elements in UI

Type: string

'owner'

Description: Database id of owner

Type: uuid

'config':

Description: Config for dashboard, encodes items and their position and size

Type: JSON Object

'visibility':

Description: Determines the share level of this dashboard

Type: Enum(Private, Shared, Semi-Public, Public)

Default: 0

'creation_time'

Description: Timestamp of when this chart was created

Type: string

'modification_time'

Description: Timestamp of when this chart was last modified

Type: string

User

'id':

Description: Database id

Type: uuid

'username':

Description: Display name

Type: string

'email':

Description: E-Mail address linked to the account, login name

Type: e-mail

'first_name':

Description: First name

Type: string

Default: ""

'last_name':

Description: Last name

Type: string

Default: ""

'real_name':

Description: Flag indicating if first_name and last_name should generally be served on lookup and used for user search

Type: boolean

Default: false

'public_profile':

Description: Flag indicating if profile should turn up in search. Private profiles are still searchable by full id

Type: boolean

Default: false

Datentypen

Zum Einlesen und Zeichnen der Visualisierungen werden grundlegende Datentypen unterschieden. Dazu ist eine Beschreibung der Beschaffenheiten wie folgt klassifiziert:

- Zahl: Float, Integer, oder ein String, der als solcher gelesen werden kann
- Text: String oder zu String konvertierbares Objekt
- Liste: Eine Liste an Daten, mit dem Datenschema als Argument
- [X, Y, ...]: 1 bis n mal X, Y wiederholt
- Breitengrad: Float zwischen -90 und +90
- Längengrad: Float zwischen -180 und +180
- Polygon: GeoJSOn Objekt (siehe https://datatracker.ietf.org/doc/html/rfc7946) [Liste(Längengrad, Breitengrad), ...]

Dies ist notwendig, um zu unterscheiden, welche Visualisierungen für eine automatisierte Verarbeitung geeignet sind. Beim Einlesen der Datenreihen wird gegen eine Reihe von Regeln geprüft, welche den Formaten im folgenden Abschnitt folgen. Bei Erfüllung dieser Regeln gilt die Datenreihe als geeignet und kann für diese Art Visualisierung gezeichnet werden. Die Visualisierungen sind vollkommen anpassbar und können wie im Abschnitt Konfigurationsmöglichkeiten anhand ihrer Eigenschaften angepasst werden.

Formate

Jede Visualisierung setzt eine bestimmte Formatierung voraus. Dies beinhaltet neben den grundlegenden Klassifizierungen der Inhalte auch die Häufigkeit der Vorkommen, sowie die Art ihrer eingehenden Sortierung (lexikographische Ordnung) sowie einer Beschränkung ihrer maximal darstellbaren Datenpunkte. Die Erfüllung aller Kriterien einer Visualisierung erlaubt das Zeichnen des Datensatzes und die Visualisierung wird in die Ergebnisliste aufgenommen.

Format	minimale Datenpunkte	maximale Datenpunkte	Ordnung nötig?	mögliche Datentypen
Liste(Zahl, Text)	0	-	nein	Barchart, Piechart
Liste(Zahl, [Zahl, Zahl, ...])	0	-	nein	Bubblechart
Liste(Zeit, [Zahl, Zahl, ...])	0	-	nein	Bubblechart
Liste(Text, Text, Zahl)	0	-	nein	ChordChart
Liste(Text, [Zahl, ...])	2	100	nein	Heatmap
Liste(Zahl, Text, Zahl, Liste(Zahl))	0	100	nein	Heatmap
Liste(Zahl, [Zahl, ...]))	0	-	Ja	LineChart, Scatterchart
Liste(Zeit, [Zahl, ...]))	0	-	Ja	LineChart, Scatterchart

Liste(Breitengrad, Längengrad, [Text, ...])	0	-	nein	POI-Karte
Liste(Polygon, Text)	0	-	nein	Polygon
Liste(Polygon, Text)	0	-	nein	Polygon
Liste(Zahl, [Zahl, ...]))	0	-	Nein	Scatterchart
Liste(Zeit, [Zahl, ...]))	0	-	Nein	Scatterchart

Konfigurationsmöglichkeiten

Zu jeder Visualisierung besteht die Möglichkeit, sie im Detail anzupassen, bevor sie gezeichnet wird und in ein Webdokument übertragen werden kann. Es werden allgemeine Eigenschaften sowie Spezielle für eine Visualisierungsart unterschieden. Die Eigenschaften sind hier nach Visualisierung aufgelöst.

Allgemein

Schlüssel	Typ	Beschreibung
t_width	Integer	Breite der Visualisierung in Pixel
t_height	Integer	Höhe der Visualisierung in Pixel
t_colors	String	Liste zu verwendende Farben für Datenreihen. Es wird der String einer Liste von JS-Farbwerten erwartet
t_line_stroke	String	Farbe von Achsenlinien als JS-Farbe
t_shape_rendering	String	Parameter für Graphenrendering. Siehe https://developer.mozilla.org/en-US/docs/Web/SVG/Attribute/shape-rendering
t_font_size	Integer	Schriftgröße von Labeln, z.B. Achsenbeschriftung in Pixel

t_label_size	Integer	Schriftgröße von Labeln, z.B. Achsenbeschriftung in Pixel
t_padding	Integer	Padding zwischen Elementen

Barchart

Schlüssel	Typ	Beschreibung
t_viewport	Integer	Maximale Anzahl gleichzeitiger Balken
t_jumplength	Integer	Anzahl der Objekte, die auf einmal gescrollt werden soll
t_verticalscale	String	Skalierungsart der Balken, mögliche Optionen sind "linear" und "log"
t_iconwidth	Integer	Icon-Weite in Pixel
t_iconheight	Integer	Icon-Höhe in Pixel
t_iconcolor	String	Farbe von Icons
t_iconhighlight	String	Farbe von Icons bei Mouseover
t_xlabel	String	X-Achsenbeschriftung
t_ylabel	String	Y-Achsenbeschriftung
t_barwidth	Integer	Balkenweite in Pixel
t_threshold	Integer	

Bubblechart

Schlüssel	Typ	Beschreibung
t_viewport	Integer	Größe des angezeigte Sichtfelds
t_jumplength	int	Scrollschrittlänge

t_scales	Liste(String,String)	Skalierung der X und Y Achse, Optionen sind "linear", "log", "pow", "date"
t_datakeys	[String, ...]	Labels für Datenreihen
t_circleopacity	Float	Deckkraft der Bubbles zwischen 0 und 1
t_highlightfactor	Float	
t_minradius	Integer	Minimaler Radius einer Bubble
t_maxradius	Integer	Maximaler Radius einer Bubble
t_xlabel	String	X-Achsenbeschriftung
t_ylabel	String	Y-Achsenbeschriftung
t_timeformat	String	D3 Zeitformatsstring
t_iconwidth	Integer	Icon-Weite in Pixel
t_iconheight	Integer	Icon-Höhe in Pixel
t_iconcolor	String	Farbe von Icons
t_iconhighlight	String	Farbe von Icons bei Mouseover

Chordchart

Schlüssel	Typ	Beschreibung
t_datakeys	[String, ...]	Labels für Datenreihen
t_textpadding	Integer	Padding für Labels
t_elementfontsizse	Integer	Fontgröße von Labels
t_tickfontsize	Integer	Fontgröße von Ticks
t_tickprefix	String	Prefix, das vor alle Ticks gesetzt wird

Heatmap

Schlüssel	Typ	Beschreibung
t_datakeys	[String, ...]	Labels für Datenreihen
t_zoom_threshold	Float	Zoomlevel, ab dem Kurnamen zu Vollnamen aufgelöst werden
t_tooltip_div_border	String	Styleinfo für die Umrandung von Tooltips
t_fill_opacity	Float	Deckkraft der Farbhervorhebung zwischen 0 und 1
t_map_stroke	String	Farbe der Kartengrenzen
t_mouseover_opacity	Float	Deckkraft der Farbhervorhebung bei Mouseover zwischen 0 und 1
t_mouseout_opacity	Float	Deckkraft der Farbhervorhebung beim Ende von Mouseover zwischen 0 und 1
t_legendborder	String	Farbe der der Legendenumrandung
t_legendwidth	Integer	Breite der Legende in Pixel
t_legendheight	Integer	Höhe der Legende in Pixel
t_legendticksize	Integer	Größe der Ticks der Legende in Pixel

Linechart

Schlüssel	Typ	Beschreibung
t_viewport	Integer	Größe des angezeigte Sichtfelds
t_jumplength	int	Scrollschrittlänge
t_scales	Liste(String,String)	Skalierung der X und Y Achse, Optionen sind "linear", "log", "pow", "date"

t_datakeys	[String, ...]	Labels für Datenreihen
t_xlabel	String	X-Achsenbeschriftung
t_ylabel	String	Y-Achsenbeschriftung
t_timeformat	String	D3 Zeitformatsstring
t_iconwidth	Integer	Icon-Weite in Pixel
t_iconheight	Integer	Icon-Höhe in Pixel
t_iconcolor	String	Farbe von Icons
t_iconhighlight	String	Farbe von Icons bei Mouseover

Piechart

Schlüssel	Typ	Beschreibung
t_datakeys	[String, ...]	Labels für Datenreihen
t_highlightfactor	Float	Deckkraft der Kuchenschlitze bei Mouseover

Point-of-Interest (POI)

Schlüssel	Typ	Beschreibung
t_datakeys	[String, ...]	Labels für Datenreihen
t_zoom_threshold	Float	Zoomlevel, ab dem Kurnamen zu Vollnamen aufgelöst werden
t_tooltip_div_border	String	Styleinfo für die Umrandung von Tooltips
t_map_fill	String	Füllfarbe für Karte

t_map_stroke	String	Randfarbe für Karte
t_mouseover_opacity	String	Deckkraft der Punkte bei Mouseover
t_fill_opacity	String	Deckkraft der Karte
t_mouseout_opacity	String	Deckkraft der Punkte beim Ende von Mouseover
t_max_poi	Integer	Maximale Punkte, die gleichzeitig angezeigt werden
t_circle_fill	String	Füllfarbe für Punkte
t_circle_stroke	String	Randfarbe für Punkte
t_circle_radius	Float	Punktgröße
t_circle_stroke_width	Float	Punktrandstärke

Polygon

Schlüssel	Typ	Beschreibung
t_datakeys	[String, ...]	Labels für Datenreihen
t_zoom_threshold	Float	Zoomlevel, ab dem Kurnamen zu Vollnamen aufgelöst werden
t_tooltip_div_border	String	Styleinfo für die Umrandung von Tooltips
t_map_fill	String	Füllfarbe für Polygone
t_map_stroke	String	Randfarbe für Karte
t_fill_opacity	String	Deckkraft der Karte
t_mouseover_opacity	String	Deckkraft der Punkte bei Mouseover
t_mouseout_opacity	String	Deckkraft der Punkte beim Ende von Mouseover
t_outer_map_fill	String	Füllfarbe für Kartenumriss

Scatterschart

Schlüssel	Typ	Beschreibung
t_viewport	Integer	Größe des angezeigte Sichtfelds
t_jumplength	int	Scrollschrittlänge
t_scales	Liste(String,String)	Skalierung der X und Y Achse, Optionen sind "linear", "log", "pow", "date"
t_datakeys	[String, ...]	Labels für Datenreihen
t_circleopacity	Float	Deckkraft der Kreise zwischen 0 und 1
t_circleradius	Integer	Kreisradius
t_xlabel	String	X-Achsenbeschriftung
t_ylabel	String	Y-Achsenbeschriftung
t_timeformat	String	D3 Zeitformatsstring
t_iconwidth	Integer	Icon-Weite in Pixel
t_iconheight	Integer	Icon-Höhe in Pixel
t_iconcolor	String	Farbe von Icons
t_iconhighlight	String	Farbe der Icons bei Mouseover