Die neue Jobfeed API

Um den Arbeitsmarkt noch transparenter zu machen, veröffentlicht Textkernel eine neue Jobfeed API. Jobfeed ist unsere Big Data Plattform mit den meisten Jobs in ganz Europa und wird nun interessierten Entwicklern zur Verfügung gestellt werden, um die Daten in eigenen Applikationen abzubilden.

Integrieren Sie Arbeitsmarktdaten in Echtzeit

Die neue Jobfeed Search-API wurde entwickelt, um interessierten Benutzern die Möglichkeit zu geben die Daten bestmöglich zu verwenden. Die RESTful API erlaubt es, Echtzeit Jobfeed Daten direkt in die eigene Anwendung zu integrieren. Daten zu spezifischen Jobs können einfach unter Bezugnahme von Filtern und Kriterien abgefragt werden. Möchten Sie zum Beispiel einen Daten-Feed von allen offenen Software Engineer Positionen in Berlin, sortiert von der Aktuellsten bis zur Letzten, live integrieren? Unsere API kann nun alle Jobs in strukturierter Form liefern, dies kann wiederum einfach in Ihrer Anwendung visualisiert werden.

Generieren Sie dynamische Datenvisualisierungen

Unsere API unterstützt Aggregationen, vorausgesetzt Suchkriterien, ein Feld zur Gruppierung und eine Metrik zur Berechnung sind gegeben. Zeigen Sie zum Beispiel monatlich den Anstieg an Sales Jobs in Deutschland oder das Durchschnittsgehalt im Gesundheitswesen in einer bestimmten Region live in Ihrer Anwendung. Jede Suchanfrage wird in JavaScript Object Notation (JSON) geliefert, die Benutzern die Möglichkeit gibt, dynamisch Diagramme, Grafiken und andere Visualisierungen zu erzeugen.

Die Jobfeed API ist in allen Jobfeed-Ländern verfügbar (dies sind zur Zeit Deutschland, Frankreich, die Niederlanden, Belgien, Österreich und UK). Zukünftige Pläne betreffend der API beziehen sich auf die Aggregation von verschiedenen Feldern, um eine stärke Analyse mit historischen Daten zu ermöglichen.

User für den Hackathon (gültig bis zum 05.06.2015)

www.jobfeed.de
User: hackathon
Passwort: hackathon2015

Jobfeed API

This is an API that allows you to search and group Jobfeed jobs. It is accessible via HTTP GET requests. You need to send your Jobfeed username and password in every request via Basic Authentication.

Searching

In order to search for jobs you should access

`https://www.jobfeed.de/api/v3/search`

You will get a JSON response consisting of an object with two properties: `total_count` (the total number of the results in the set) and `results`(an array of job objects). By default you will get a maximum of 10 job objects per request.

Filtering

For some fields that have codes as values you get in the response not only the code, but also a string description of the code. For example:

```
[…]
"education_level": {
    "value": 8,
    "label": "Description of code 8 here"
}
[…]
```

When filtering by a certain education level you'd use the value (8 in the example above), not the label.

You can turn off labels completely by adding a `_labels=0` parameter to the URL. That would make the example above look like:`"education_level": 8`.

In order to filter the results you would use GET parameters. For example, to get all the jobs for which the `profession` field is 227 you would access:

`https://www.jobfeed.de/api/v3/search?profession=227`

If you want to pass multiple values for the same field (for example, say you want to look for jobs for which profession is either 227 or 4980) you have to suffix the field name with []:

https://www.jobfeed.de/api/v3/search?profession[]=227&profession[]=4980

You can set filters on multiple fields. For example, to get all the jobs for which profession is 4980 and education_level is 2 you can access:

https://www.jobfeed.de/api/v3/search?profession=4980&education_level=2

To filter on a boolean field (for example, via_intermediary) you can pass 1 to represent true and 0 to represent false:

https://www.jobfeed.de/api/v3/search?via_intermediary=0

For the profession and location fields value normalization is also performed. That means that instead of the code you can use a string value. For example:

https://www.jobfeed.de/api/v3/search?profession=PHP Programmer

or

https://www.jobfeed.de/api/v3/search?location=Den Haag

You should never filter on location_name or location_coordinates. The results may be unexpected if you do. Always use locationinstead.

Advanced filtering

You can filter based on the existence or non-existence of a field by using the special values _exists and _not_exists:

https://www.jobfeed.de/api/v3/search?conditions_description=_exists

For the location field you can also filter using a radius around a central point by suffixing the field name with __radius. For example, to search 25 kilometers around Amsterdam you would access:

https://www.jobfeed.de/api/v3/search?location__radius=Amsterdam__25

or using our code for Amsterdam (1000):

https://www.jobfeed.de/api/v3/search?location__radius=1000__25

You can also use actual geographical coordinates instead of a city:

https://www.jobfeed.de/api/v3/search?location__radius=52.08,4.32__10

You can do a range search on a field by suffixing the field name with __range. For example, to search for jobs for which education_level is between 3 and 8 (inclusive) you would access:

https://www.jobfeed.de/api/v3/search?education_level__range=3__8

You can specify open-ended ranges by omitting either the start or the end (but not both): 3__ (equal or greater than 3) or __8 (equal or less than 8).

For date ranges (date and expiration_date fields) you can use calculated values. To search for jobs that have the date field between 1 year ago and now you can use:

https://www.jobfeed.de/api/v3/search?date__range=now-1y__now

Units supported are y (year), M (month), w (week), d (day), h (hour), m (minute) and s (second). Each expression has to start with an anchor date which can be either now, or a date suffixed with ||. More examples: now-1h, now+1h+1m, 2014-03-01||+1w.

You can do an expression search on a field by suffixing the field name with __exp. This would allow you, among other things, to:

- use boolean expressions like: /search?full_text__exp=(transport OR auto) AND diesel
- use phrase searches: /search?full_text__exp="compleet aanbod" – the words need to be right next to each other
- use fuzzy searches: /search?full_text__exp=colour~ – it will match "color" too; you can specify an acceptable edit distance like:colour~2; a better match means a higher rank for the document
- use proximity searches: /search?full_text__exp="junior manager"~3 – similar to the fuzzy search, but on a word level; the closer the words are to each other the higher the rank
- use wildcard searches: /search?full_text__exp=auto*

NB: you can only use one of __radius, __range, __exp at a time. You can't do something like location__radius__range__exp.

You can negate any filter by suffixing it with __not. For example, you can look for jobs that don't have education_level 5:

https://www.jobfeed.de/api/v3/search?education_level__not=5

You can also negate radius, range or expression searches:

https://www.jobfeed.de/api/v3/search?education_level__range__not=3__8

NB: In order to improve readability we didn't urlencode the parameters in this guide. You must do it in your application (or, better yet, build the URLs using a library that does it automatically) because otherwise the service will see a value like now+1h as now 1h and return an error.

Control parameters

You can use some special parameters in order to control what is being returned. All the control parameters start with an underscore to distinguish them from the filters.

To get a certain number of results (other than the default of 10) you can use the _limit parameter:

https://www.jobfeed.de/api/v3/search?_limit=100

To skip a certain number of jobs from the top of the result set you can use _offset. Say you want to skip the first 50 jobs and start from the 51st:

https://www.jobfeed.de/api/v3/search?_offset=50

You can use _limit and _offset together in order to paginate the results.

You can use _sort (with a field name as a value) and _sortdir (either asc or desc) to order the results:

https://www.jobfeed.de/api/v3/search?_sort=date&_sortdir=desc

If you want to get a subset of the fields you can use the _fields parameter:

https://www.jobfeed.de/api/v3/search?_fields=date,job_title,profession

Grouping

You do grouping by accessing:

`https://www.jobfeed.de/api/v3/aggregate`

If you access the URL above directly you'll get an error message. All the requests to this endpoint must have a `_group` parameter which specifies the field to group by:

`https://www.jobfeed.de/api/v3/aggregate?_group=profession`

Filtering

Filtering works exactly as it does for the `/search` endpoint:

`https://www.jobfeed.de/api/v3/aggregate?_group=profession&location=Amsterdam&education_level__range=3__8`

Control parameters

We already mentioned the `_group` parameter which has to be present in all the `/aggregate` requests. The value will be a field name.

You can limit the number of "buckets" using `_limit`.

`https://www.jobfeed.de/api/v3/aggregate?_group=profession&_limit=100`

By default you get the document count for each bucket, but you can change that using the `_metric` parameter. The value passed to this parameter should be in the `operation__field` format. The supported operations are `min`, `max`, `sum` and `avg`:

`https://www.jobfeed.de/api/v3/aggregate?_group=profession&_metric=avg__salary`

There is one exception to the `operation__field` rule: `_metric=count_postings` will return counts of job postings (ie: the duplicate postings are all included).

You can use `_sort` and `_sortdir` and they behave similarly to the `/search` counterparts. For grouping sorting by a field value doesn't usually have much sense (although you can do it if you must) which is why we provide two special values that you can pass as `_sort`:

_group which sorts by the group name and _value which sorts by the aggregated value (job count, or the specified metric).

https://www.jobfeed.de/api/v3/aggregate?_group=profession&_metric=avg__salary&_sort=_value&_sortdir=desc

You can use the _labels parameter and it has the same meaning as for /search.

If you group by a date field (like date and expiration_date) you can suffix the field name by __week, __month, __quarter or __yearin order to group by that time interval:

https://www.jobfeed.de/api/v3/aggregate?_group=date__month

NB: If you group by a time interval _limit, _sort and _sortdir won't have any effect.

Fields

You can get a list of the available fields by accessing

https://www.jobfeed.de/api/v3/fields

Fields with a fixed list of values will have a possible_values property that enumerats those values.

By default you get only the fields you have access to. If you want to get all the fields you can use the _get_all parameter:

https://www.jobfeed.de/api/v3/fields?_get_all=1

Miscellaneous

For every request you can have a _pretty=1 parameter that will cause the output JSON to be pretty-printed. Do not use this in production.