nuPlanner

by Al Johri, Sai Kolli

Package
controllers

controllers
Class Application

```
java.lang.Object
    │
    +-Controller
        │
        +-controllers.Application
```

public class **Application**
extends Controller

Constructors

Application

public **Application**()

Methods

events

public static Result **events**()

The events controllers grabs a list of events from the database based on three parameters: start, end, and query. The start and end parameters are unixtimestamps (in seconds) while the query paramter is a string used to filter the events.

This controller adheres to the specifications laid out by fullCalendar, the javascript Calendar used in the view. fullCalendar: http://arshaw.com/fullcalendar/docs/event_data/events_json_feed/

This controller is called by the /events url

Returns:
Return a JSON string of events matching the input parameters.

controllers
Class Facebook

```
java.lang.Object
     │
     +-controllers.Facebook
```

---

public final class **Facebook**
extends java.lang.Object

The Facebook class contains ACCESS_TOKENS, FQL query fragments, graph query fragments, and helper functions to encapsulate the RestFB query methods.

Author:
> Al Johri

Fields

APP_ACCESS_TOKEN

public static final java.lang.String **APP_ACCESS_TOKEN**

> Constant value: **524073037656113|l1aTC3FhsPHJEeRZfWB9vk70nAk**

USER_ACCESS_TOKEN

public static final java.lang.String **USER_ACCESS_TOKEN**

> Constant value:
> **CAAHcpEMZB8DEBANQkZAyhJCXEYnAn2UW9paEZC7VWGJCb9a1ouGismLu8xKfCZCZBdtAl3Ld4iVC6u2Ha
> XHqmknogy4RMuMswUMqc17PKSwAt4UmQoqDT4RIHKQaeHpnoXjXkCWTFlx7fNKKvDLUaauyS3KIuFsvziJ
> aCphDuCgZDZD**

CENTER_LATITUDE

public static final java.lang.String **CENTER_LATITUDE**

> Constant value: **42.054581**

CENTER_LONGITUDE

public static final java.lang.String **CENTER_LONGITUDE**

> Constant value: **-87.677192**

CENTER_RADIUS

public static final java.lang.String **CENTER_RADIUS**

> Constant value: **1000**

FQL_EVENT_FIELDS

public static final java.lang.String **FQL_EVENT_FIELDS**

> Constant value: **eid, name, creator, host, start_time, end_time, location, venue, description, pic, pic_big, pic_cover, pic_small, pic_square, timezone, update_time, all_members_count, attending_count, declined_count, unsure_count, not_replied_count**

FQL_EVENT_MEMBER_FIELDS

public static final java.lang.String **FQL_EVENT_MEMBER_FIELDS**

> Constant value: **eid**

FQL_PLACE_FIELDS

public static final java.lang.String **FQL_PLACE_FIELDS**

> Constant value: **page_id**

FQL_EVENT_TABLE

public static final java.lang.String **FQL_EVENT_TABLE**

> Constant value: **event**

FQL_EVENT_MEMBER_TABLE

public static final java.lang.String **FQL_EVENT_MEMBER_TABLE**

> Constant value: **event_member**

FQL_PLACE_TABLE

public static final java.lang.String **FQL_PLACE_TABLE**

> Constant value: **place**

FQL_LIMIT

public static final java.lang.String **FQL_LIMIT**

> Constant value: **LIMIT 51000**

FQL_ORDER

public static final java.lang.String **FQL_ORDER**

> Constant value: **ORDER BY start_time ASC**

GRAPH_EVENT_FIELDS

`public static final java.lang.String` **`GRAPH_EVENT_FIELDS`**

> Constant value:
> **`id,name,owner,description,start_time,end_time,location,venue,privacy,updated_time,`**
> **`picture,ticket_uri`**

GRAPH_SEARCH_TYPE

`public static final java.lang.String` **`GRAPH_SEARCH_TYPE`**

> Constant value: **`event`**

GRAPH_LIMIT

`public static final java.lang.String` **`GRAPH_LIMIT`**

> Constant value: **`5000`**

BLOCKED_ORGANIZATIONS

`public static final java.lang.String` **`BLOCKED_ORGANIZATIONS`**

BLOCKED_VENUES

`public static final java.lang.String` **`BLOCKED_VENUES`**

Constructors

Facebook

`public` **`Facebook`**`()`

Methods

facebook_fql_query

`public static java.util.List` **`facebook_fql_query`**`(java.lang.String token,`
`        java.lang.String query)`

> [facebook_fql_query description]

> **Parameters:**
> > `token` - [description]
> > `query` - [description]

> **Returns:**
> > [description]

facebook_graph_query

```
public static com.restfb.json.JsonObject facebook_graph_query(java.lang.String token,
        java.lang.String q,
        java.lang.String type,
        java.lang.String fields,
        java.lang.String center,
        java.lang.String distance,
        java.lang.String limit)
```

[facebook_graph_query description]

Parameters:
token - [description]
q - [description]
type - [description]
fields - [description]
center - [description]
distance - [description]
limit - [description]

Returns:
[description]

```
public static com.restfb.json.JsonObject facebook_graph_query(java.lang.String token,
```

controllers
Class Scrape

```
java.lang.Object
   |
   +-Controller
       |
       +-controllers.Scrape
```

Direct Known Subclasses:
> [ScrapeFacebook](#), [ScrapeWildcatConnection](#)

---

public class **Scrape**
extends Controller

This class serves as a wrapper for ScrapeFacebook and ScrapeWildcatConnection classes.
Author:
> Al Johri
Version:
> 0.1

Constructors

Scrape

```
public Scrape()
```

Methods

scrape

```
public static Result scrape()
```

> This method runs smaller scrape methods from the Scrape classes's children in the necessary order to get the best final results.

> ScrapeFacebook.scrape_locations() must be run last because it uses previously scraped events to generate its queries.

> This controller is called by the /scrape url

> **Returns:**
> > Return success message.

controllers
Class ScrapeFacebook

```
java.lang.Object
    │
    +-Controller
        │
        +-controllers.Scrape
            │
            +-controllers.ScrapeFacebook
```

public class **ScrapeFacebook**
extends Scrape

The ScrapeFacebook class scrapes events from Facebook using the Graph API and Facebook Query Langauge. It stores the resulting events to the database.
Author:
  Al Johri

Constructors

ScrapeFacebook

public **ScrapeFacebook**()

Methods

scrape_graph

public static Result **scrape_graph**()

  Scrapes events from Facebook using the Facebook Graph API. It searches for events radially outward from predefined center coordinates with a predefined radius.

  This method requires a USER_ACCESS_TOKEN as opposed to the standard APP_ACCESS_TOKEN. Thus, this method can only be run upon user authentication. Thus, this method is currently not being used in the "scrape sequence," the sequence of functions run to scrape all events (defined in Scrape.java).

  TODO: Enable this function after implementing Facebook User Authentication.

  Returns:
    Return a JSON string of all scraped events.

scrape_fql

public static Result **scrape_fql**()

Scrapes events from Facebook using the Facebook Query Language (FQL). It searches for events using the defined FQL_QUERY.

The FQL_QUERY searches for events that are created by Facebook Pages. It filters these events based on pages whose location is within CENTER_RADIUS meters from (CENTER_LATITUDE, CENTER_LONGITUDE).

As stated above, this query only searches for events that are created by a page. The event_member table in FQL contains uids and page_ids of, as the name suggests, members of events. The logic behind the query is that if a page is a member of an event, the page is the creator the event. Most pages have a location associated with them. If the location is within CENTER_RADIUS meters from (CENTER_LATITUDE, CENTER_LONGITUDE) the event will be returned.

However, one major pitfall is international/national organizations that have their headquarters within this radius. The page resides here, but the actual event may be in Washinton, DC or another country. Some examples of these are SAE and Rotary International. There is no obvious solution for filtering with these events, especially if they don't have a listed venue. The filter_events function in Utilities.java explores this in greater detail.

Returns:
Return a JSON string of all scraped events.

---

scrape_organizations

```
public static Result scrape_organizations()
```

Scrapes organizations using seed data in the MyOrganization model. It searches for events using an FQL query that finds all events created by a specific organization.

For some reason, the majority (if not all) of events found using this method are from past years. However, the locations, venues, and other data that result are valuable for other functions.

Returns:
Return a JSON string of all scraped events.

---

scrape_locations

```
public static Result scrape_locations()
```

Scrape Locations uses data already in the MyEvent table to search for events. It uses the getAllEventLocations function in MyEvent.java to find a list of all locations where events currently reside based on the events in the database. It then uses the Facebook Graph API to perform a text based query of these locations. This is a powerful function and grabs a myriad of events. However, filtering is extremly important as the majority of these events will be outside of the city/state/country.

Returns:
Return a JSON string of all scraped events.

controllers
Class ScrapeWildcatConnection

```
java.lang.Object
    |
    +-Controller
        |
        +-controllers.Scrape
            |
            +-controllers.ScrapeWildcatConnection
```

---

public class **ScrapeWildcatConnection**
extends Scrape

The ScrapeWildcatConnection class scrapes events within WildcatConnection and stores them to the database.
Author:
> Al Johri

Constructors

ScrapeWildcatConnection

public **ScrapeWildcatConnection**()

Methods

scrape_wildcatconnection

public static Result **scrape_wildcatconnection**()

controllers
Class Seed

```
java.lang.Object
    │
    +-Controller
        │
        +-controllers.Seed
```

public class **Seed**
extends Controller

The Seed class collects data from various sources on the web that will be saved to the database in order to aid scrapers.

Author:
> Sai Praneeth

Constructors

Seed

public **Seed**()

Methods

seed

public static Result **seed**()

> Calls all seed methods. Currently just seed_organizations().
>
> This controller is called by the /seed url
>
> **Returns:**
> > Return success message.

seed_locations

public static Result **seed_locations**()

> Get list of buildings at Northwestern from maps.northwestern.edu. Evanston: http://maps.northwestern.edu/dialog/building_list/Evanston/ Chicago: http://maps.northwestern.edu/dialog/building_list/Chicago/ TODO: implement this function
>
> **Returns:**
> > Return success message.

seed_organizations

public static Result **seed_organizations**()

> Get html pages for all organizations from northwestern collegiatelink Parse these pages to get their facebook id Add these id's to graph url to get their facebook info Parse this info to get necessary details of that organization Save this info in database into My Organization table

Returns:
Return success message.

Returns:
Return success message.

controllers
Class Utilities

```
java.lang.Object
   |
   +-Controller
        |
        +-controllers.Utilities
```

public class **Utilities**
extends Controller

The Seed class collects data from various sources on the web that will be saved to the database in order to aid the scrapers.

Constructors

Utilities

public **Utilities**()

Methods

createEvent

public static [MyEvent](#) **createEvent**(com.restfb.json.JsonObject event)

> Input Event Parse event to get all its properties as in MyEvent Constructor
>
> **Returns:**
> > MyEvent constructor

saveOrUpdate

public static boolean **saveOrUpdate**([MyEvent](#) event)

> Input event Save event in Database
>
> **Returns:**
> > true if added else false

filter_event

public static boolean **filter_event**([MyEvent](#) event)

> input event Filter events based on some fixed blocked venues and organizations Also Filter events based on their city
>
> **Returns:**
> > true if event is not filtered else false

(continued from last page)

save_events

```
public static java.util.List save_events(java.util.List events)
```

Input events list For all events in list which are in json create MyEvent object Filter events add filtered events which are MyEvent objects to eventlist

Returns:
    eventlist

save_events

```
public static java.util.List save_events(com.restfb.json.JsonArray events)
```

Input events array For all events in array convert them to jsonObject For all events in list which are in json create MyEvent object Filter events add filtered events which are MyEvent objects to eventlist

Returns:
    eventlist

Package
models

models
Class MyEvent

```
java.lang.Object
    |
    +-Model
        |
        +-models.MyEvent
```

public class **MyEvent**
extends Model

Fields

eid

public java.lang.Long **eid**

name

public java.lang.String **name**

creator

public java.lang.String **creator**

location

public java.lang.String **location**

venue

public java.lang.String **venue**

start_time

public DateTime **start_time**

end_time

public DateTime **end_time**

pic

`public java.lang.String` **`pic`**

pic_small

`public java.lang.String` **`pic_small`**

pic_big

`public java.lang.String` **`pic_big`**

pic_square

`public java.lang.String` **`pic_square`**

description

`public java.lang.String` **`description`**

find

`public static <any>` **`find`**

findLong

`public static <any>` **`findLong`**

Constructors

MyEvent

```
public MyEvent(java.lang.Long eid,
               java.lang.String name,
               java.lang.String creator,
               DateTime start_time,
               DateTime end_time,
               java.lang.String location,
               java.lang.String venue,
               java.lang.String description,
               java.lang.String pic,
               java.lang.String pic_small,
               java.lang.String pic_big,
               java.lang.String pic_square)
```

Methods

toJSONString

```
public java.lang.String toJSONString()
```

getAllEventLocations

```
public static java.util.ArrayList getAllEventLocations()
```

models
Class MyOrganization

```
java.lang.Object
    |
    +-Model
        |
        +-models.MyOrganization
```

public class **MyOrganization**
extends Model

Fields

fbid

public java.lang.Long **fbid**

name

public java.lang.String **name**

creator

public java.lang.String **creator**

url

public java.lang.String **url**

location

public java.lang.String **location**

description

public java.lang.String **description**

find

public static <any> **find**

findLong

```
public static <any> findLong
```

Constructors

MyOrganization

```
public MyOrganization(java.lang.Long fbid,
                      java.lang.String name,
                      java.lang.String location,
                      java.lang.String url,
                      java.lang.String description)
```

Methods

getJsonObject

```
public java.lang.String getJsonObject()
```

models
Class Person

```
java.lang.Object
    │
    +-Model
        │
        +-models.Person
```

Direct Known Subclasses:
> [Student](#)

---

public class **Person**
extends Model

Fields

id

```
public java.lang.Long id
```

---

name

```
public java.lang.String name
```

Constructors

Person

```
public Person(java.lang.String name)
```

models
Class Student

```
java.lang.Object
    │
    +-Model
        │
        +-models.Person
            │
            +-models.Student
```

public class **Student**
extends Person

Fields

find

```
public static <any> find
```

Constructors

Student

```
public Student(java.lang.String name)
```

Index

T

U

V