JUST A CAT —— AN INFORMATION APPLICATION

1. Introduction

Just a Cat is an informative Android application. It integrates latest news from *The New York Times*, *The Guardian* and some mainstream medias in China, weather information, projects on github trending, a turing robot which can chat with users (support voice input) and translation (support voice input).

2. APIs

The Guardian

> 《卫报》（The Guardian）是英国的全国性综合内容日报，与《泰晤士报》、《每日电讯报》被合称为英国三大报。该报注重报道国际新闻，擅长发表评论和分析性专题文章。
>
> *The Guardian* is a British daily newspaper which is focus on reporting international news. It is good at making comments and analytical thematic articles. The paper's readership is generally on the mainstream left of British political opinion.

The New York Times (need to use proxy)

> *The New York Times* is an American newspaper based in New York City with worldwide influence and readership. It has long been regarded within the industry as a national "newspaper of record".

Integration of some mainstream medias in China (Tencent, CCTV, World Wide) (unofficial)

> All of them are large Chinese news websites.

Turing Robot

> An open intelligence chat robot platform which provide APIs for users to chat with turing robot.

Weather (unofficial)

> Provide the weather information of every province, including weather forecast, air quality and life advice.

Bing pictures (background picture of weather page) (unofficial)

> Update a beautiful picture everyday.

Youdao Translator

> A Translator.

Speech recognition method developed by iFly (SDK is under `Just/app/libs`)

> Support voice input.

Github Trending (unofficial, Code)

> Exhibits some potential open source projects on Github.

3. Implement

- Adopt material design style, use the slide menu to switch different information pages.
- Use retrofit to make Http&Https requests, use GSON and SimpleXml to parse data.

Data parsing

- Deal with XML
  - `GithubParser Class` Use SimpleXml to parse XML data from API of github trending

```
public class GitHubParser
{

    @ElementList(inline = true,entry = "project",required = false)
    private ArrayList<Project> projects;

    public ArrayList<Project> getProjects() {
        return projects;
    }

    public void setProjects(ArrayList<Project> projects) {
        this.projects = projects;
    }

    @Root(name = "project",strict = false)
```

```java
    public static class Project
    {
        @Element(name = "owner",required = false)
        private String owner;

        @Element(name = "repository_name",required = false)
        private String repository_name;

        @Element(name = "name",required = false)
        private String name;

        @Element(name = "descriptions",required = false)
        private String descriptions;

        @Element(name = "language",required = false)
        private String language;

        @Element(name = "stars",required = false)
        private String stars;

        @Element(name = "url",required = false)
        private String url;

        @ElementList(entry = "contributor",inline = true,required =
false)
        private ArrayList<Contributor> contributors;

        public String getDescriptions()
        {
            if (descriptions == null)
                return "This article does not have a title, please
click to view details";
            return descriptions;
        }

        public void setDescriptions(String descriptions)
        {
            this.descriptions = descriptions;
        }

        public String getTitle()
        {
            if (descriptions == null)
                return "This article does not have a title, please
click to view details";
            return descriptions;
        }

        public String getUrl()
        {
            if (url == null) return "https://github.com/";
            return url;
```

```java
        }

        public void setUrl(String url)
        {
            this.url = url;
        }

        public ArrayList<Contributor> getContributors()
        {
            return contributors;
        }

        public void setContributors(ArrayList<Contributor>
contributors)
        {
            this.contributors = contributors;
        }

        public String getName()
        {
            if(name == null) return "None";
            return name;
        }

        public void setName(String name)
        {
            this.name = name;
        }

        public String getLanguage()
        {
            if(language == null) return "None";
            return language;
        }

        public void setLanguage(String language)
        {
            this.language = language;
        }

        public String getStars()
        {
            return stars;
        }

        public void setStars(String stars)
        {
            this.stars = stars;
        }

        public String getOwner()
        {
```

```java
            return owner;
        }

        public void setOwner(String owner)
        {
            this.owner = owner;
        }

        public String getRepository_name()
        {
            return repository_name;
        }

        public void setRepository_name(String repository_name)
        {
            this.repository_name = repository_name;
        }

        @Root(name = "contributor", strict = false)
        public static class Contributor
        {
            @Element(name = "avatar",required = false)
            private String avatar;

            public String getAvatar()
            {
                return avatar;
            }

            public void setAvatar(String avatar)
            {
                this.avatar = avatar;
            }
        }
    }

    public static class Repository
    {
        @SerializedName("subscribers_count")
        private String watchers_count;

        @SerializedName("forks_count")
        private String forks_count;

        public String getWatchers_count()
        {
            return watchers_count;
        }

        public void setWatchers_count(String watchers_count)
        {
            this.watchers_count = watchers_count;
```

```java
        }

        public String getForks_count()
        {
            return forks_count;
        }

        public void setForks_count(String forks_count)
        {
            this.forks_count = forks_count;
        }
    }
}
```

- Deal with JSON example
    - ○ `GuardianJsonParser Class` Use GSON to parse JSON data form API of The Guardian

```java
public class GuardianJsonParser
{
    // Field constants
    private static final String JSON_RESPONSE = "response";
    private static final String JSON_RESULTS = "results";
    private static final String JSON_TAGS = "tags";
    private static final String JSON_BLOCKS = "blocks";
    private static final String JSON_REQUESTED_BODY_BLOCKS =
"requestedBodyBlocks";
    private static final String JSON_BODY_LATEST = "body:latest";
    private static final String JSON_ELEMENTS = "elements";
    private static final String JSON_ASSETS = "assets";
    private static final String JSON_FILE = "file";
    private static final String VALUE_NO_IMAGE = "No Image";
    private static final String VALUE_NO_AUTHOR = "No Author";
    private static final String ARTICLE_ID = "id";
    private static final String ARTICLE_TYPE = "type";
    private static final String ARTICLE_SECTION_ID = "sectionId";
    private static final String ARTICLE_SECTION_NAME = "sectionName";
    private static final String ARTICLE_WEB_PLUBLICATION_DATE =
"webPublicationDate";
    private static final String ARTICLE_WEB_TITLE = "webTitle";
    private static final String ARTICLE_WEB_URL = "webUrl";
    private static final String ARTICLE_API_URL = "apiUrl";
    private static final String ARTICLE_IS_HOSTED = "isHosted";
```

```java
    private static ArrayList<Articles> mArticlesArrayList;
    // GuardianJsonParser Constructor
    public GuardianJsonParser()
    {
    }

    public static ArrayList<Articles> getArticlesFromJSON(String
articlesJSONString)
    {
        try
        {
            // Create the articlesJSON object with the
articlesJSONString parameter
            JSONObject jsonObject = new
JSONObject(articlesJSONString);
            JSONObject responseObject =
jsonObject.getJSONObject(JSON_RESPONSE);
            JSONArray resultsArray =
responseObject.getJSONArray(JSON_RESULTS);
            int articlesQty = resultsArray.length();
            mArticlesArrayList = new ArrayList<>();
            // Loop through the articlesArray to parse each Json
object needed
            for (int i = 0; i < articlesQty; i++)
            {
                JSONObject articleRecord =
resultsArray.getJSONObject(i);
                // Parse the individual data elements needed
                String articleId =
articleRecord.getString(ARTICLE_ID);
                String articleType =
articleRecord.getString(ARTICLE_TYPE);
                String articleSectionId =
                    articleRecord.getString(ARTICLE_SECTION_ID);
                String articleSectionName =
                    articleRecord.getString(ARTICLE_SECTION_NAME);
                String articleWebPublicationDate =

 articleRecord.getString(ARTICLE_WEB_PLUBLICATION_DATE);
                String articleWebTitle =
articleRecord.getString(ARTICLE_WEB_TITLE);
                String articleWebUrl =
articleRecord.getString(ARTICLE_WEB_URL);
                String articleApiUrl =
articleRecord.getString(ARTICLE_API_URL);
                Boolean articleIsHosted =
                    articleRecord.getBoolean(ARTICLE_IS_HOSTED);
                // Try to get the author from the tags
                JSONArray articleTagsArray =
articleRecord.getJSONArray(JSON_TAGS);
                String articleAuthor = VALUE_NO_AUTHOR;
                try
```

```java
                    {
                        JSONObject articleContributor =
                            articleTagsArray.getJSONObject(0);
                        articleAuthor =
articleContributor.getString(ARTICLE_WEB_TITLE);
                    } catch (JSONException ignored)
                    {
                    }
                    // Try to get the image
                    String articleFile = VALUE_NO_IMAGE;
                    try
                    {
                        JSONObject articleBlocks =
                            articleRecord.getJSONObject(JSON_BLOCKS);
                        JSONObject articleRequestedBlocks =

 articleBlocks.getJSONObject(JSON_REQUESTED_BODY_BLOCKS);
                        JSONArray articleBodyLatestArray =

 articleRequestedBlocks.getJSONArray(JSON_BODY_LATEST);
                        JSONObject articleBodyLatest =
                            articleBodyLatestArray.getJSONObject(0);
                        JSONArray articleElements =

 articleBodyLatest.getJSONArray(JSON_ELEMENTS);
                        JSONObject articleAssets =
articleElements.getJSONObject(1);
                        JSONArray articleImageArray =
                            articleAssets.getJSONArray(JSON_ASSETS);
                        JSONObject articleImage =
articleImageArray.getJSONObject(3);
                        articleFile = articleImage.getString(JSON_FILE);
                    }
                    catch (JSONException ignored)
                    {
                    }
                    // Create an Article Object
                    Articles article = new Articles(articleId,
articleType,
                                                    articleSectionId,
                                                    articleSectionName,

 articleWebPublicationDate,
                                                    articleWebTitle,
articleWebUrl,
                                                    articleApiUrl,
articleIsHosted,
                                                    articleAuthor,
articleFile);
                    // Add the Article Object to the ArrayList
                    mArticlesArrayList.add(article);
                }
```

```
        }
        catch (JSONException e)
        {
            Log.e("Exception: ", "" + e);
            e.printStackTrace();
        }
        return mArticlesArrayList;
    }
}
```

- Utility Class Use GSON to parse JSON data from weather API

```
public class Utility
{
    public static boolean handleProvinceResponse(String response)
    {
        if (!TextUtils.isEmpty(response))
        {
            try
            {
                JSONArray allProvinces = new JSONArray(response);
                for (int i = 0; i < allProvinces.length(); i++)
                {
                    JSONObject provinceObject =
allProvinces.getJSONObject(i);
                    Province province = new Province();

 province.setProvinceName(provinceObject.getString("name"));

 province.setProvinceCode(provinceObject.getInt("id"));
                    province.save();
                }
                return true;
            }
            catch (JSONException e)
            {
                e.printStackTrace();
            }
        }
        return false;
    }

    public static boolean handleCityResponse(String response, int
provinceId)
    {
        if (!TextUtils.isEmpty(response))
        {
```

```java
            try
            {
                JSONArray allCities = new JSONArray(response);
                for (int i = 0; i < allCities.length(); i++)
                {
                    JSONObject cityObject =
allCities.getJSONObject(i);
                    City city = new City();
                    city.setCityName(cityObject.getString("name"));
                    city.setCityCode(cityObject.getInt("id"));
                    city.setProvinceId(provinceId);
                    city.save();
                }
                return true;
            }
            catch (JSONException e)
            {
                e.printStackTrace();
            }
        }
        return false;
    }

    public static boolean handleCountyResponse(String response, int
cityId)
    {
        if (!TextUtils.isEmpty(response))
        {
            try
            {
                JSONArray allCounties = new JSONArray(response);
                for (int i = 0; i < allCounties.length(); i++)
                {
                    JSONObject cityObject =
allCounties.getJSONObject(i);
                    County county = new County();

 county.setCountyName(cityObject.getString("name"));

 county.setWeatherId(cityObject.getString("weather_id"));
                    county.setCityId(cityId);
                    county.save();
                }
                return true;
            }
            catch (JSONException e)
            {
                e.printStackTrace();
            }
        }
        return false;
    }
```

```java
    public static Weather handleWeatherResponse(String response)
    {
        try
        {
            JSONObject jsonObject = new JSONObject(response);
            JSONArray jsonArray
 =jsonObject.getJSONArray("HeWeather");
            String weatherContent =
jsonArray.getJSONObject(0).toString();
            return new Gson().fromJson(weatherContent,
Weather.class);
        }
        catch (Exception e)
        {
            e.printStackTrace();
        }
        return null;
    }
}
```

Dependencies

```
    compile 'com.lguipeng.bubbleview:library:1.0.0'
    compile 'com.github.medyo:android-about-page:1.2.4'
    compile 'org.litepal.android:core:1.4.1'
    compile 'com.android.support:design:26.+'
    compile 'de.hdodenhof:circleimageview:2.1.0'
    compile 'com.android.support:recyclerview-v7:26.+'
    compile 'com.android.support:cardview-v7:26.+'
    compile 'com.github.bumptech.glide:glide:3.7.0'
    compile 'com.squareup.okhttp3:okhttp:3.4.1'
    compile 'com.squareup.retrofit2:retrofit:2.0.1'
    compile 'com.squareup.retrofit2:converter-gson:2.0.1'
    compile 'com.google.code.gson:gson:2.7'
    compile 'com.youth.banner:banner:1.4.9'
    compile 'com.roughike:bottom-bar:2.3.1'
    compile 'com.github.GrenderG:Toasty:1.3.0'
    compile 'com.squareup.picasso:picasso:2.5.2'
    compile 'com.github.bumptech.glide:glide:3.5.2'
    compile 'io.reactivex:rxandroid:1.1.0'
    compile 'io.reactivex:rxjava:1.1.3'
    compile 'com.squareup.retrofit:retrofit:2.0.0-beta1'
    compile 'com.squareup.retrofit2:retrofit:2.0.0-beta4'
    compile 'com.squareup.retrofit:converter-gson:2.0.0-beta2'
    compile 'com.squareup.retrofit:adapter-rxjava:2.0.0-beta2'
    compile 'com.astuetz:pagerslidingtabstrip:1.0.1'
    compile 'com.facebook.fresco:fresco:0.9.0+'
    compile ('com.squareup.retrofit2:converter-simplexml:2.1.0'){
```

```
        exclude group: 'xpp3', module: 'xpp3'
        exclude group: 'stax', module: 'stax-api'
        exclude group: 'stax', module: 'stax'
    }
    compile 'io.reactivex.rxjava2:rxjava:2.1.6'
    compile 'com.squareup.retrofit2:adapter-rxjava2:2.3.0'
    compile 'io.reactivex.rxjava2:rxandroid:2.0.1'
```

4.Other functions

- Change Themes

  In fact, this function just change the color of toolbar from blue to grey.

- Favorites

  You can touch the "favorites" button on news page to add it to your favorites list, you can also clean your favorites list.

5. Development and configuration

Language: Java

IDE: IntelliJ IDEA

How to run: Download the project and run in IDEA

About APIs: You must provide your own keys of following five APIs in order to connect with and fetch data from them.

- Turing Robot

  Create a API on http://www.tuling123.com and put your key and user id here in `Activity/TulingActivity.java`

```java
    // 请求数据
    //@param text 输入框的发送数据
    private void request(String text)
    {
//        把输入的文本数据存储在请求实体类中
        Ask ask = new Ask();
        Ask.UserInfoBean info = new Ask.UserInfoBean();
        info.setApiKey("put in your tuling_api_key");
        info.setUserId("put in your tuling_user_id");
```

- Youdao Translator

  Create a API on https://www.xfyun.cn and and put your key and app id here in

  `Activity/TranslateActivity.java`

  ```
  //点击按钮方法
  public void translateString(View view)
  {
      String appId = "put in your youdao_appid";
      String appKey = "put in your youdao_api_key";
  ```

- iFly

  Create a API on https://ai.youdao.com and download SDK. Put your app id here in

  `Activity/TranslateActivity.java`

  ```
  private void initSpeech()
  {
      SpeechUtility.createUtility( context: this,  s: SpeechConstant.APPID+"=put in your ifly_app_id");
  }
  ```

- The New York TImes

  Creat Create a API on https://developer.nytimes.com and put your key here in

  `util/NewsQueryUtils.java`

  ```
  // Main method that has sub-routines to fetch top stories
  public static List<News> fetchNews(String newsCategory)
  {
      // Create valid url object from the requestURL
      String requestUrl = "https://api.nytimes.com/svc/topstories/v2/%1$s.json?api-key=%2$s";
      String fullUrl = String.format(requestUrl, newsCategory, "put in your newyorktimes_api_key");
      URL url = createUrl(fullUrl);
  ```

- The Guardian

  Creat Create a API on https://developer.nytimes.com and put your key here in
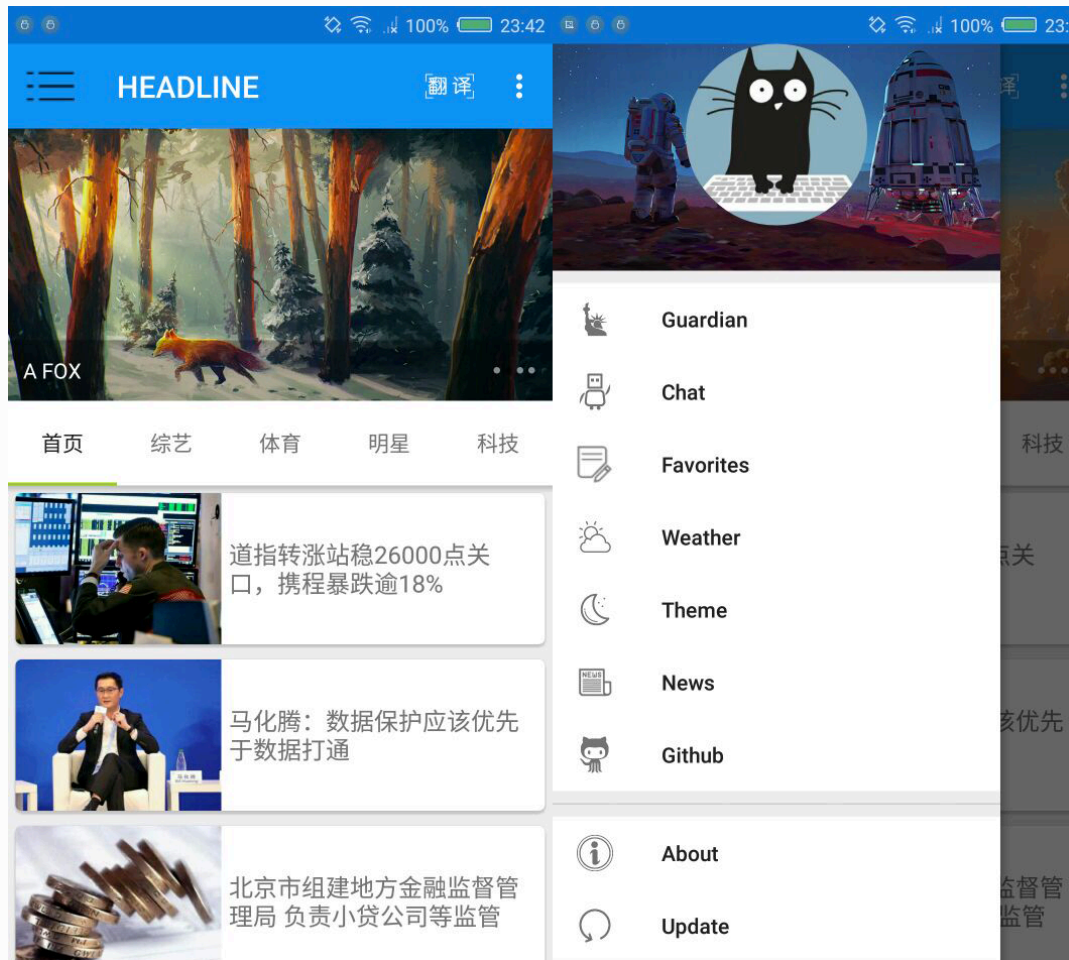
  `Data/API.java`

  ```
  // An API class to connect to the Guardian and get the results
  public final class API
  {
      //private static final String API_KEY = BuildConfig.GUARDIAN_API_KEY;
      private static final String API_KEY = "put in your guardian_api_key";
  ```

6. ScreenShots

Home page and slide menu

Click the menu button on the top left corner to expanding the slide menu.

Favourites

Click the favorites button on the bottom right corner to add this news into your favorites list.
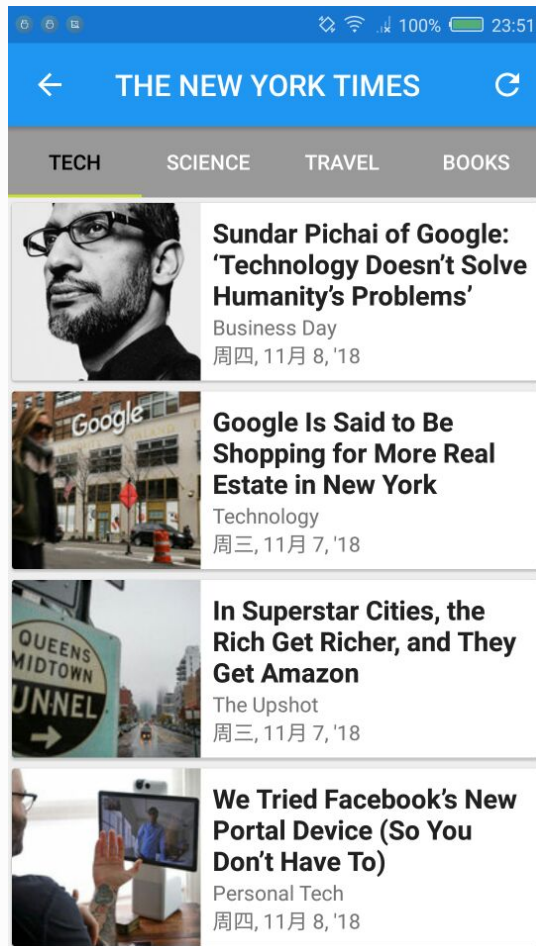Click the delete button on the op right corner to clean your favorites list.

Weather

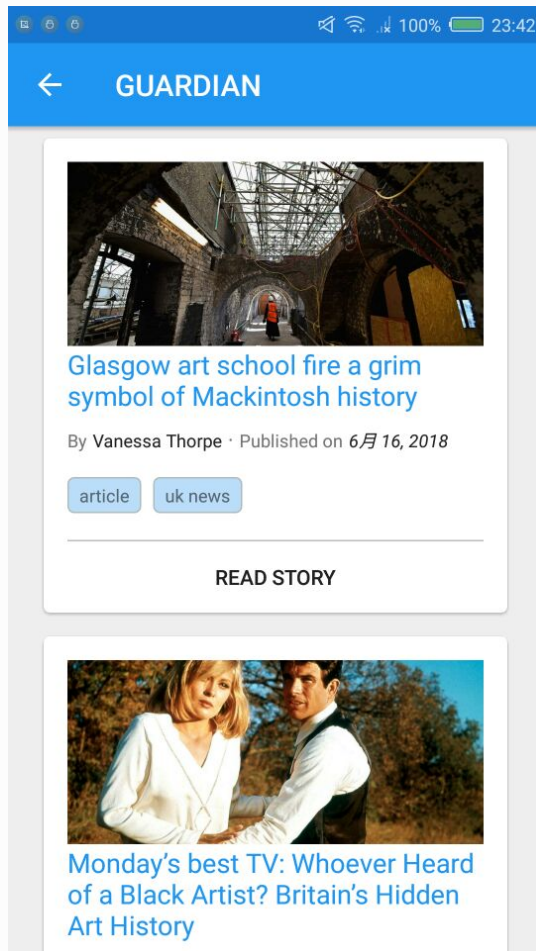Click the menu button on the top left corner to expanding the provinces and cities menu.
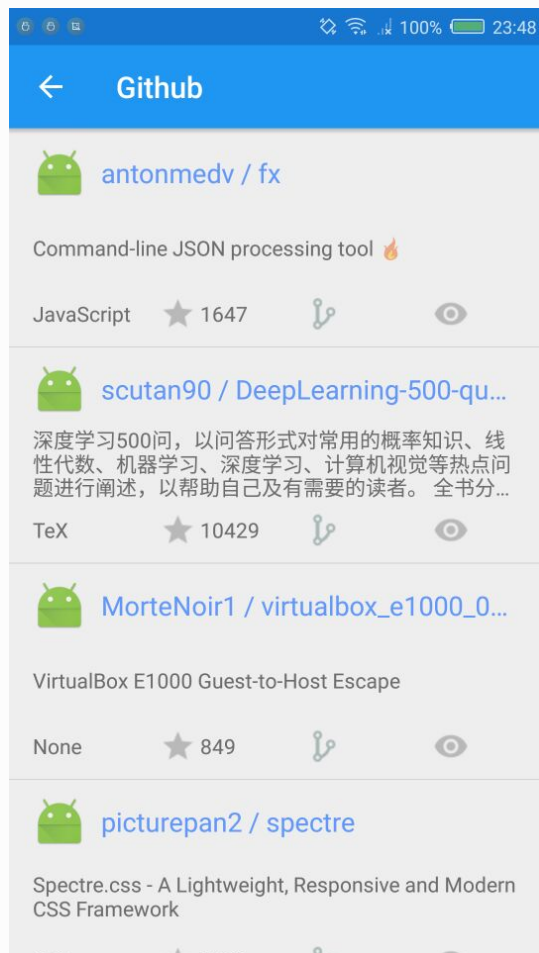
The New York Times

The Guardian

图灵机器人

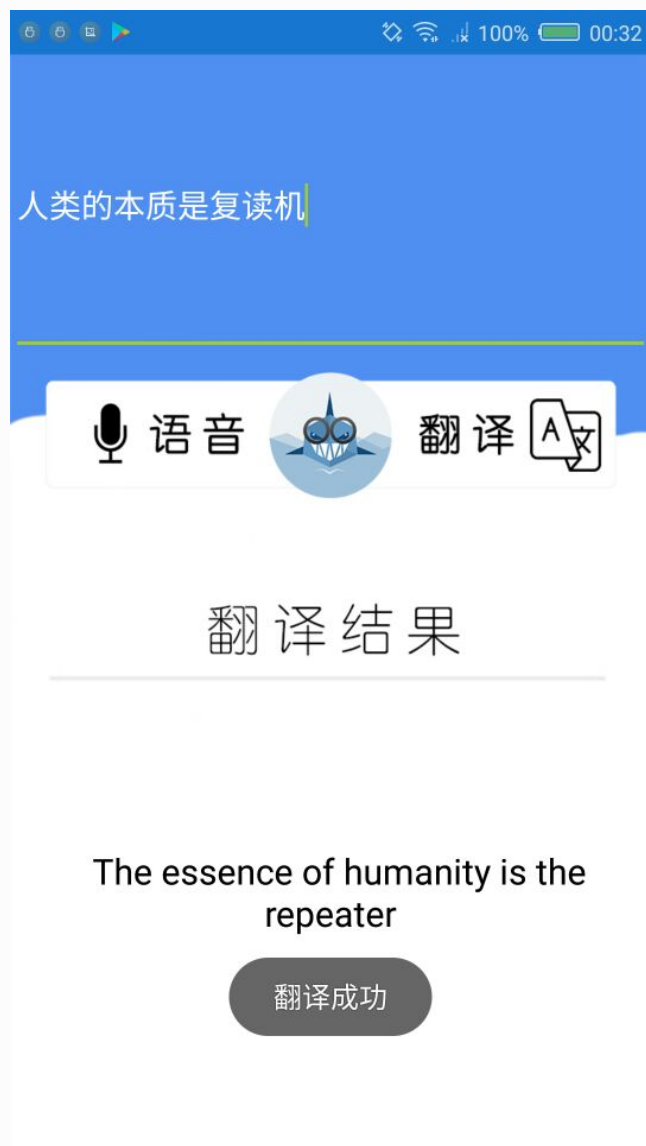Click the "microphone" button to change your voice to words.

Github

Translator

Click the "microphone" button to change your voice to words.

人类的本质是复读机

语音　翻译

翻译结果

The essence of humanity is the repeater

翻译成功

Change theme
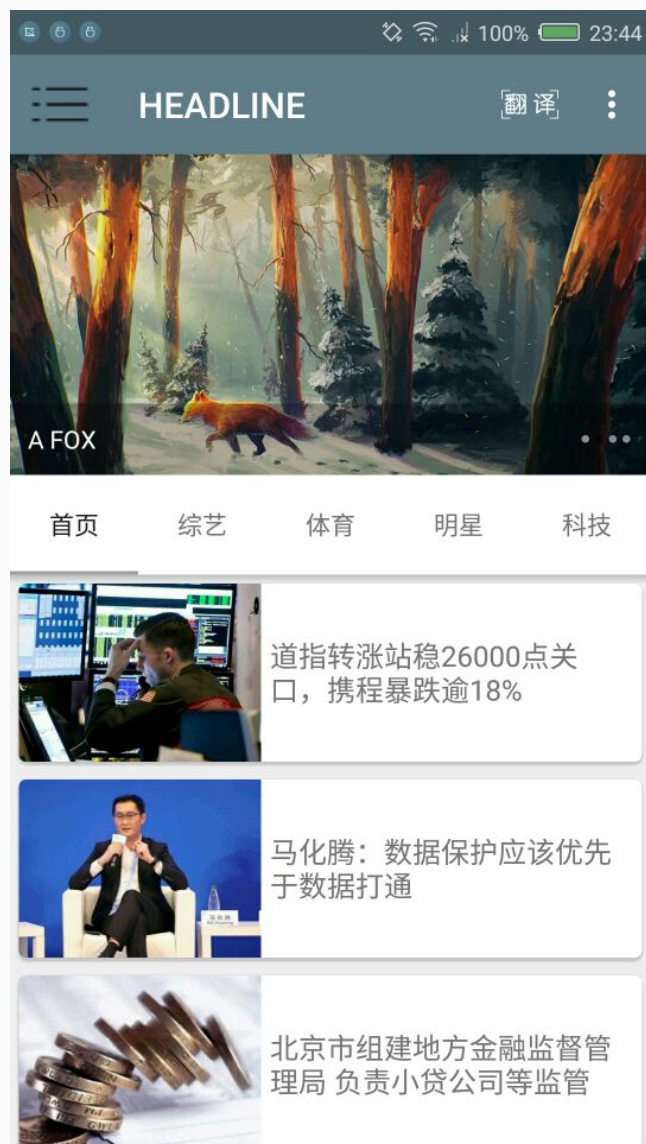
About

Just a Cat

只是一只懂得很多的猫

News, Article, Weather, Translate, Technology
and Chatting