

Description of Github discourse sample for import into DiscourseDB

For Aug 25th, 2016 Hack day
Chris Bogart

Discussion on github: Owner/projects -> issues -> comments

This screenshot shows the GitHub repository page for 'eneim / Toro'. The repository has 407 commits and is on the 'develop' branch. The 'Issues' tab is selected, showing 15 open issues. The repository description is 'Video list auto playback ma...'. The repository structure includes files like '.idea', 'art', 'gradle/wrapper', and 'ref'.

This screenshot shows a filtered view of the issues for 'eneim / Toro'. The filters are set to 'is:issue is:'. The issues list shows 15 open issues and 17 closed issues. The first three issues are visible:

- Autoplay does not** #81 opened 2 days ago by [user]
- UI hangs when ne** #62 opened on Jun 20 by chetanurmaliya
- Extension for Video Playback Controller or Enhancement** #61 opened on Jun 12 by eneim

This screenshot shows the details of issue #62, 'UI hangs when network is slow (2G)', opened by chetanurmaliya on Jun 20. The issue has 2 comments.

Issue Title: UI hangs when network is slow (2G) #62

Issue Description: I am using com.github.eneim:Toro:1.2.0. In side the RecycleView list we have the videos for auto play. When network is slow (like in 2G) entire RecycleView list UI hangs. I am not able to scroll the list. I see lot of updates in repository. Is library has updated version. Please let me know the best approach to handle it.

Comments:

- chentanurmaliya** commented on Jun 20: I am using com.github.eneim:Toro:1.2.0. In side the RecycleView list we have the videos for auto play. When network is slow (like in 2G) entire RecycleView list UI hangs. I am not able to scroll the list. I see lot of updates in repository. Is library has updated version. Please let me know the best approach to handle it.
- eneim** (Owner) commented on Jun 20: @chetanurmaliya thanks for issue report. Sadly I could not test my lib under 2G connection, I will try it with some dev config later. One of the suggestion now is to disable the auto playback on low network connection. Please search for network state confirmation and then see how I temporary disable Toro or even permanently disable Toro. Hope this help. I'm actively working on ver 2 of this lib, which will shift to ExoPlayer as playback backend. Hopefully I could release it soon. So please stay tune.

Format of data

I've extracted some issue data in the form of CSV files containing comments, events, dates, and users; also references to outside URLs and other issues. One file per issue, plus an extra commits.csv file for commits and other non-issue project information.

Header row:

- `rectype,issueid,project_owner,project_name,actor,time,text,action,title,provenance,paths,plus_1,urls,issues,userref,code`

Example data rows:

`issue_title,4,eneim,Toro,eneim,2016-02-02 23:21:27+00:00,"Need to stop a video which ...",start issue,Single video`

`playback,api.github.com,,false,[],[],[]`

`issue_comment,4,eneim,Toro,eneim,2016-02-03 06:40:11+00:00,Fixed in #7 ,,,api.github.com,,false,[],["{"raw": "#7", "refstyle": "#d", "parts": ["eneim", "Toro", "7", ""]}"],[],[]`

Interpretation:

- This is issue #4 in eneim's project called "Toro". User eneim posed the issue, then commented on it
- Issue title and text, and comment text, are all bolded above
- The text "#7" in the comment is a reference to another issue, so the "issues" column has parsed this out.

rectype column values in the Github extract

issue_event, pull_request_history: When issues or pullrequests are merged, closed, opened, etc. Just a timestamp, not much useful data

issue_title: Issue creation date, with title and description

issue_comment: additional comments on issues

pull_request_commit_comment: The comments attached to commits within a pull request.

pull_request_comment: Code review comments in pull requests

issue_crossref, pull_request_crossref: Duplicates issue_comment, in cases where text refers to another issue; but it is labeled by the repo and issue number that is referred *to*, and has the reference info in parseable json format

commit_message: What people type when they check in code

commit_comment: Further comments also attached to a commit later

readme: Contents of the latest commit of the project's designated README file

Columns in Github extract (p. 1 of 2)

rectype: Described on previous slide

issueid: The user-friendly issue or PR number (not internal database index). If this is 0, the file contains project-global things, like commits and readme files

project_owner, project_name: If these are X and Y, they refer to repo <http://github.com/X/Y>. project_owner could be a person or an organization.

actor: The username of the person who posted the text. May or may not match project_owner.

time: In format YYYY-MM-DD hh:mm:ss+00:00

text: What they wrote, or “None” or “missing comment”

action: What kind of event for issue_event or pull_request_history: e.g. opened, merged, etc.

Columns in Github extract (p. 2 of 2)

title: Issue title (in issue_title rows), description of referrer (in issue_crossref)

provenance: ghtorrent or githubarchive: where I got the data from

parts: For commits only, this says which files were edited

plus_1: TRUE if the comment begins with “+1”

urls: JSON-formatted list of URLs referenced in the text

issues: JSON-formatted parsing of cross-reference; e.g. [{"raw": "#10", "refstyle": "#d", "parts": ["Yelp", "swagger_spec_validator", "10", ""]}]

userref: JSON-formatted parsing of user name reference e.g. ["@dnephin"]

code: JSON-formatted list of languages that are literally quoted in the text, e.g. ["python"] if the text has ```python

When doing the exercise, don't forget:

Check out <http://github.com/discoursedb/discoursedb-github-workshop>

Create a custom.properties file, and substitute your own name in the database name

Set a dataset name and directory (src/main/resources/data) as Run Configuration arguments

Fill in TutorialConverterService.java and TutorialConverter.java

Data format is csv, but some columns are embedded JSON.

<http://tutorials.jenkov.com/java-json/jackson-objectmapper.html>

Hints! Random lines of code from the solution

```
@Autowired private DiscoursePartService discoursePartService;
Contribution contrib = contributionService.createTypedContribution(ContributionTypes.THREAD_STARTER);
contrib.setStartTime(currentComment.getTime());
discoursePartService.addContributionToDiscoursePart( ...
Discourse curDiscourse = discourseService.createOrGetDiscourse("Github");
DiscoursePart issueDP = discoursePartService.createOrGetTypedDiscoursePart(..., DiscoursePartTypes.GITHUB_ISSUE);
import edu.cmu.cs.lti.discoursedb.core.model.user.User;
import edu.cmu.cs.lti.discoursedb.core.service.macro.ContentService;
import edu.cmu.cs.lti.discoursedb.tutorial.model.GitHubIssueComment;
public long mapIssueComment(GitHubIssueComment currentComment) {
switch (currentComment.getRectype()) {
```