Code Story: Information Tracking for Program Comprehension

Felix Grund and Nick Bradley
Department of Computer Science
University of British Colombia
Vancouver, BC, Canada
ataraxie,nbrad11@cs.ubc.ca

ABSTRACT

abstract.tex

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous; D.2.8 [**Software Engineering**]: Metrics—*complexity measures, performance measures*

General Terms

Theory

Keywords

ACM proceedings, LaTeX, text tagging

1. INTRODUCTION

introduction.tex

2. RELATED WORK

related-work.tex

3. APPROACH

approach.tex

4. EVALUATION

evaluation.tex

5. IMPLEMENTATION

The implementation of CodeStory consists of three parts that interact with each other. There is (1) a Chrome extension responsible for collecting meta-information from StackOverflow when snippets are copied, (2) a backend receiving these information and persisting them in a database and (3) an Atom package that extends pasted snippets with a link to a page showing these meta information. This section provides a brief description of each part.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
Copyright 20XX ACM X-XXXXX-XX-X/XX/XX ...$15.00.

5.1 Chrome Extension

The CodeStory Chrome extension mainly consists of a Content Script that is executed whenever a user visits a StackOverflow page. An event listener for the browser's *copy event* is added and henceforth called whenever content from the page is copied to the clipboard.

The listener first collects the required meta-information from the current page an saves them in an *storyData* object. The fields that are collected are shown in Table 1. To be able to identify a *storyData* object, a unique ID is created with a hash based on the current timestamp and the selected text. The created hash and the *storyData* object are now sent to the CodeStory backend with a POST request. Measures to ensure acceptance of these cross-origin requests are taken on the backend and the Chrome extension may thus send simple Ajax requests.

The missing link is now the one to the Atom package: the pasted snippet must be somehow associated with its meta-information. For that purpose, the clipboard content is enriched with the created hash which is achieved using the JavaScript clipboard API[1]. The Atom package may then extract this hash from the clipboard content upon paste and proceed accordingly.

5.2 Backend

XXX

5.3 Atom Package

XXX

6. DISCUSSION

discussion.tex

7. FUTURE WORK

future-work.tex

8. CONCLUSION

conclusion.tex

9. REFERENCES

[1]https://developer.mozilla.org/en-US/docs/Web/API/ClipboardEvent

Table 1: Fields collected as meta-information from StackOverflow upon copy

Field Name	Description
copiedFrom	Whether the snippet was copied from a question or an answer.
type	Whether the copied snipped is code-only.
originalSelection	Exact text that was selected in the browser upon copy.
questionTitle	Title of the question of the current page.
questionUrl	URL of the question (i.e. the URL of the current page).
questionContent	Full content of the question as text.
questionContentWithHtml	Full content of the question preserving the HTML markup.
questionVotes	Number of votes the question received.
answerUrl*	URL of the answer.
answerContent*	Full content of the answer as text.
answerContentWithHtml*	Full content of the answer preserving the HTML markup.
answerVotes*	Number of votes the answer received.
accepted*	Whether the answer was accepted.
accessTime	Timestamp when the page was accessed.
fullCodeSnippet	The full code snippet (only available if the snippet is part of a code snippet).

* Only available if copied from an answer.