EDIT - task #6778

remove big objects from taxeditor git repository

07/06/2017 09:12 AM - Andreas Kohlbecker

Status:	New	**Start date:**	
Priority:	Priority14	**Due date:**	
Assignee:	Andreas Kohlbecker	**% Done:**	0%
Category:	devOps	**Estimated time:**	0:00 hour
Target version:	Reviewed Next Major Release		
Severity:	normal		

Description

beim Taxeditor gibt es ein Problem, das die Migration zu github verhindert:

```
remote: error: File eu.etaxonomy.taxeditor.webapp/lib/cdmlib-remote-webapp.war is 101.81 MB; this exceeds GitHub's file size limit of 100.00 MB
```

vom cdmlib-remote-webapp.war gibt es einige git Objekte die zu groß sind:

```
49293516 taxeditor-updateSite/plugins/eu.etaxonomy.cdmLibrary_2.1.0.84.jar
72778150 eu.etaxonomy.taxeditor.webapp/lib/cdmlib-remote-webapp.war
72781304 eu.etaxonomy.taxeditor.cdmlib/src/main/resources/etc/jetty/cdmlib-remote-webapp.war
82123402 eu.etaxonomy.taxeditor.webapp/target/eu.etaxonomy.taxeditor.webapp-4.0.0-SNAPSHOT.jar
106108206 eu.etaxonomy.taxeditor.cdmlib/src/main/resources/etc/jetty/cdmlib-remote-webapp.war
106112416 eu.etaxonomy.taxeditor.cdmlib/src/main/resources/etc/jetty/cdmlib-remote-webapp.war
106112431 eu.etaxonomy.taxeditor.cdmlib/src/main/resources/etc/jetty/cdmlib-remote-webapp.war
106750525 eu.etaxonomy.taxeditor.webapp/lib/cdmlib-remote-webapp.war
106751423 eu.etaxonomy.taxeditor.webapp/lib/cdmlib-remote-webapp.war
108253934 eu.etaxonomy.taxeditor.cdmlib/src/main/resources/etc/jetty/cdmlib-remote-webapp.war
108529965 eu.etaxonomy.taxeditor.webapp/src/main/resources/etc/jetty/cdmlib-remote-webapp.war
110075559 eu.etaxonomy.taxeditor.cdmlib/src/main/resources/webapp/cdmlib-remote-webapp.war
114200590 eu.etaxonomy.taxeditor.remoting/src/main/resources/etc/jetty/cdmlib-remote-webapp.war
136716323 eu.etaxonomy.taxeditor.cdmlib/src/main/resources/etc/jetty/cdmlib-remote-webapp.war
136721477 eu.etaxonomy.taxeditor.cdmlib/src/main/resources/etc/jetty/cdmlib-remote-webapp.war
136722821 eu.etaxonomy.taxeditor.cdmlib/src/main/resources/etc/jetty/cdmlib-remote-webapp.war
136739217 eu.etaxonomy.taxeditor.cdmlib/src/main/resources/etc/jetty/cdmlib-remote-webapp.war
136760622 eu.etaxonomy.taxeditor.cdmlib/src/main/resources/etc/jetty/cdmlib-remote-webapp.war
150450654 eu.etaxonomy.taxeditor.cdmlib/src/main/resources/etc/jetty/cdmlib-remote-webapp.war
157115390 eu.etaxonomy.taxeditor.cdmlib/src/main/resources/etc/jetty/cdmlib-remote-webapp.war
159246416 eu.etaxonomy.taxeditor.cdmlib/src/main/resources/etc/jetty/cdmlib-remote-webapp.war
```

Diese werden von github alle NICHT akzeptiert.

Es gibt die Möglichkeit diese Objekte im git repo zu löschen aber dadurch werden alle betroffenen Branches neu geschrieben, ich denke dass sich dabei auch die Commit-Hashes ändern. Die Verknüpfungen von Commits zu Redmine Tickets und Wikiseiten würden dabei kaputt gehen. Wir sollten uns also nach einer anderen Möglichkeit umsehen oder einen Weg finden wie kaputte commit links repariert werden können.

History

#1 - 07/06/2017 09:12 AM - Andreas Kohlbecker

Hallo Andreas,

weiß nicht, ob ihr bei dem Problem schon weiter seid? Soweit ich das sehe, würden aber bei allen drei Optionen die Commit-Hashes geändert und somit die bisherigen Links ungültig, oder nicht? Das würde ich in der Tat auch als Problem sehen. Folgende zwei Lösungsmöglichkeiten würden mir einfallen:

1) Wohin gehen die bisherigen Links denn genau? Falls sie an eine URL unter Deiner Kontrolle gehen, wäre es ja ggf. eine Option dort ein Redirect-Skript laufen zu lassen, das die alten Commit-Hashes zu den neuen übersetzt (also die bisherigen URLs zu den entsprechenden GitHub-URLs mit neuen Commit-Hashes per HTTP redirect umleitet). Das wäre ja einfach zu machen, sofern man eine Datenbank hat, die alt nach neu zuordnet.

Wie würde es denn genau ablaufen, wenn man die Commit History, z.B. bei Option 2 ändert? Müsstest Du dazu ein eigenes Programm schreiben, dass jeden Commit vom bisherigen Server untersucht und dann einen entsprechend veränderten Commit an GitHub schickt oder gibt es da

vordefinierte Möglichkeiten? Hier wäre es dann interessant, ob bei diesem Schritt leicht eine entsprechende Zuordnungstabelle erstellt werden kann oder nicht.

2) Eine weitere denkbare Option wäre auch, dass man den bisherigen Git-Server beibehält und ein entsprechendes GitHub-Repository nur als Spiegel (z.B. mit der LSP-Option) benutzt, sofern das mit den vorhandenen großen Dateien technisch mit vertretbarem Aufwand möglich ist. Dann könnte es uns ggf. egal sein, dass die Commit-Hashes dort anders sind, da immer nur die am BGBM Server entscheidend sind und irgendwo verlinkt werden. Pull-Requests u.ä. von GitHub bekommt ja ja ggf. trotzdem leicht zum "Hauptserver" zurück und wir würden nach wie vor mit dem URLs unseres eigenen "Hauptservers" arbeiten. (Eine ähnliche Lösung haben wir für bioinfweb auch in Planung, um die Repositories der Öffentlichkeit über GitHub zugänglich zu machen, diese aber trotzdem bei uns kontrolliert liegen zu haben. Es wird aber noch dauern, bis das soweit ist.)

Bessere Optionen fallen mir im Moment leider auch nicht ein.

VG
Ben

#2 - 07/06/2017 09:12 AM - Andreas Kohlbecker

Hallo Ben,

zu 1)
die Commit-Hash-Links befinden sich alle in Redmine, also in den Texten der Ticketbeschreibungen, den Kommentaten, Wikiseiten und zudem gibt es noch Relationen in der Datenbank zwischen Issues und den Tabellen in denen Redmine Repositories indiziert.
Man könnte also ein Radmine-addon schreiben, dass die alten commit hashes auf die neuen mappt.

Beim recherchieren zu diesem Thema bin ich auf diesen Forumsbeitrag zum BFG Repocleaner gestoßen:
http://howtoprogram.eu/question/n-a.48972https://rtyley.github.io/bfg-repo-cleaner/

Dieser macht genau anscheinen das was wir brauchen, entfernt zu große Objekte , schreibt also die History um, speichert die alten Commit hashes als Footer der Commit-Message: Former-commit-id:

Wir bräuchten also "nur" ein redmine plugin das diesen Footer auswertet. Vielleicht hat das schon mal jemand geschrieben?

zu 2)
die LSP-Option kenne ich noch nicht, werde ich mir aber anschauen.

#3 - 07/06/2017 09:12 AM - Andreas Kohlbecker

Hallo Andreas,

1) das Redmine-Plugin wäre eine Möglichkeit, die ja im Prinzip das gleiche wie ein URL-Redirector macht. Welche von beiden nun leichter zu implementieren ist kann (mangels Detailkenntnissen von Redmine) nicht sagen. Ein URL-Redirector wäre sicherlich nicht sehr aufwendig und hätte ggf. den Vorteil, dass alle alten URLs prinzipiell gültig bleiben (und damit auch in Redmine nicht zwangsläufig geändert werden müssen) und somit auch alle Links, die evtl irgendwo anders noch existieren (in Wikis, persönlichen Notizen, Daten von Dritten oder was auch immer) auch weiterhin funktionieren würden. Ab jetzt könnte man dann ja in Redmine trotzdem die neue URLs verwenden. Will gar nicht zwangsläufig von einem Redmine Plugin statt einem URL-Redirector abraten, wollte nur noch mal auf diese Vorteile eines Redirectors hinweisen.

2) LSP kenne ich im Detail auch nicht. Grundsätzlich ist es wohl einfach eine Möglichkeit die großen Dateien woanders zu speichern. Am schönsten wäre es natürlich (egal ob LSP oder ein anderer Speicher verwendet wird), wenn das neue Repository wenigstens Links zu den entfernten Dateien statt diesen selbst enthalten würde, z.B. einfach in Form einer Textdatei, die nur einen Hinweistext auf die Größenbeschränkung und eine URL unter der die bisherige Datei zu finden ist enthält. Weiß nicht, ob Repocleaner oder auch das LSP-System von GitHub dazu bereits automatisierte Möglichkeiten bieten. Falls ja, wäre es sicher schön die zu nutzen. Falls dazu aber auch wieder eigene Plugins benötigt würden, wäre es sicher abhängig von der Frage, wie wichtig es ist, alte Zustände des Repositories vollständig rekonstruieren zu können.

VG
Ben

#4 - 07/06/2017 09:12 AM - Andreas Kohlbecker

Hallo Ben,.

1)
die Commit links in Redmine sind zunächst reine markdown syntax, z.B: commit:cdm-vaadin|05ce260 Nur wenn redmine den Commit 05ce260 im Git repository cdm-vaadin kennt, wird beim Interpretieren des Markdown ein HTML Link daraus. Ein URL-Redirector wie du ihn vorgeschlagen hast würde voraussetzen, dass die HTML Links immer generiert werden. Dies würde wiederum bedeuten, dass man ein neues Addon entwickeln oder bestehende Funktionalität in Redmine erweitern müsste.

Viele Grüße
Andreas

#5 - 07/06/2017 09:13 AM - Andreas Kohlbecker

Hallo,

kann man eigentlich eine Mapping Tabelle erstellen (altes Commit-neues Commit)?
Da ließe sich doch relativ leicht ein skript schreiben, welches die alten Commits anpasst in Redmine.

Fände ich besser, als eine Redirect-Lösung die dann dauerhaft supported werden muss.

Viele Grüße,
Andreas M.

#6 - 07/06/2017 09:13 AM - Andreas Kohlbecker

Meinst du mit anpassen, dass die Einträge der alten Commits durch neue ersetzt werden, also auf Datenbank Ebene?
Das ist auf jeden Fall auch eine Lösung. Folgende Bereiche des Redmine Datenmodells sind betroffen:

- Texten der Ticketbeschreibungen
- den Texten Issue-Kommentare,
- Wikiseiten
- Relationen zwischen Issues und den Tabellen in denen Redmine Repositories indiziert

Viele Grüße
Andreas

#7 - 07/06/2017 09:15 AM - Andreas Kohlbecker

> Meinst du mit anpassen, dass die Einträge der alten Commits durch neue ersetzt werden, also auf Datenbank Ebene?

Ja, das war die Idee.

#8 - 07/06/2017 09:16 AM - Andreas Kohlbecker

Wie gesagt, das ist im Prinzip eine gute Idee.
Wie aufwendig die Umsetzung dieser Lösung im Vergleich zu einem Addon ist müssen wir mal sehen.
Wir könnten dafür entweder eine Stored-procedure schreiben oder vielleicht ist ein rake script sogar einfacher weil Übersichltlicher in der Umsetzung, wegen der Debug Möglichkeiten und dem Logging.

Andreas

#9 - 07/06/2017 09:16 AM - Andreas Kohlbecker

Hi,

ja wie man das skripted ist letztlich egal, würde ich sehen, welche Sprache für das Problem am leichtesten ist. Ist ja nur ne einmalige Sache, kann also quick, dirty und unhandlich sein, so lang es funktionert.
Aus Java heraus ließe sich das sicherlich auch sehr schnell implementieren (evtl. schneller als mit den anderen Sprachen, da wir uns mit Java wahrscheinlich am besten auskennen).

Das Hauptproblem bei dem Skript sehe ich eigentlich daran, dass die commits ja in unterschiedlicher Länge vorhanden sein können, also von ca. 6 Zeichen bis alle Zeichen, das müsste man also mitberücksichtigten. Ansonsten ist vermutlich straight forward, wenn man weiß in welchen Tabellenfeldern die Texte vorkommen können. Aber hast du ja schon eine Liste erstellt.
Haben wir da schon ein Ticket für?

Und wenn wir irgendeine Stelle vergessen, ist es auch nicht so kritisch. So lange man die Mappingtabelle irgendwo speichert, kann man das ja noch nachholen oder, wenn dann wirklich einmaliger Nachschlagebedarf ist, dort direkt nachschauen.

Viele Grüße,
Andreas M.

#10 - 07/21/2017 10:27 AM - Andreas Müller

- Target version changed from Release 4.9 to Release 4.10

#11 - 09/24/2017 01:20 AM - Andreas Müller

- Target version changed from Release 4.10 to Release 4.11

#12 - 11/08/2017 11:15 AM - Andreas Müller

- Target version changed from Release 4.11 to Release 4.12

#13 - 12/05/2017 02:17 PM - Andreas Müller

- Target version changed from Release 4.12 to Release 4.13

#14 - 01/31/2018 12:12 PM - Andreas Müller

- Target version changed from Release 4.13 to Release 4.14

#15 - 02/15/2018 03:00 PM - Andreas Müller

- Target version changed from Release 4.14 to Release 5.0

#16 - 05/16/2018 11:19 AM - Andreas Kohlbecker

- Target version changed from Release 5.0 to Release 5.1

#17 - 06/28/2018 12:19 AM - Andreas Müller

- Target version changed from Release 5.1 to Release 5.2

#18 - 08/17/2018 03:55 PM - Andreas Kohlbecker

- Target version changed from Release 5.2 to Release 5.3

#19 - 08/28/2018 10:57 AM - Andreas Kohlbecker

Eine weitere Möglichkeit ist der Umzug auf Gitlab.com.

"To celebrate today's good news we've permanently raised our storage limit per repository on GitLab.com from 5GB to 10GB. As before, public and private repositories on GitLab.com are unlimited, **don't have a transfer limit** and they include unlimited collaborators." (https://about.gitlab.com/2015/04/08/gitlab-dot-com-storage-limit-raised-to-10gb-per-repo/)

#20 - 08/28/2018 01:39 PM - Patrick Plitzner

Andreas Kohlbecker wrote:

> Eine weitere Möglichkeit ist der Umzug auf Gitlab.com.
>
> "To celebrate today's good news we've permanently raised our storage limit per repository on GitLab.com from 5GB to 10GB. As before, public and private repositories on GitLab.com are unlimited, **don't have a transfer limit** and they include unlimited collaborators." (https://about.gitlab.com/2015/04/08/gitlab-dot-com-storage-limit-raised-to-10gb-per-repo/)

transfer limit VS file size limit. Oben in der Beschreibung steht, dass es an der Dateigröße liegt, oder?

#21 - 09/06/2018 11:18 AM - Andreas Kohlbecker

- Target version changed from Release 5.3 to Release 5.4

#22 - 09/06/2018 11:21 AM - Andreas Kohlbecker

- Tags set to git

#23 - 10/23/2018 01:52 PM - Andreas Kohlbecker

- Target version changed from Release 5.4 to Release 5.5

#24 - 01/30/2019 11:31 AM - Andreas Kohlbecker

- Target version changed from Release 5.5 to Release 5.6

#25 - 02/22/2019 05:36 PM - Andreas Kohlbecker

- Priority changed from New to Priority14

#26 - 02/22/2019 05:36 PM - Andreas Kohlbecker

- Target version changed from Release 5.6 to Reviewed Next Major Release