Vorlesungshomepages ab FS 2017

Andreas Steiger, Manuel Lüthi

1. Februar 2017

1 Allgemeines

Im Zuge der ETH-weiten Umstellung für Websiten wurde das Content Management System (CMS) Silva per Herbstsemester 2016 archiviert. Das neue CMS der ETH ist jedoch nicht darauf ausgelegt, jedes Semester dutzende neue Autoren einzuarbeiten. Deshalb musste für die Vorlesungshomepages, die bisher auch in Silva integriert waren, eine neue Lösung her.

Am D-MATH sind im Laufe der Jahre in der Gruppe um Norbert Hungerbühler, Alexander Caspar und Heinz Rasched und den Projekten Nemesis und Echo diverse Softwarelösungen entstanden, die einen Teil der Funktionalität der bisherigen Vorlesungshomepages übernehmen können. In den letzten Monaten ist nun ein System entstanden, das die Einfachheit und Offenheit von Silva mit den bestehenden Komponenten verbinden kann.

Ansehen kann man sich das Resultat auf `https://metaphor.ethz.ch/`. Diese Seite liefert grundsätzlich einfache statische HTML-Seiten aus, die über die Versionierungssoftware `git` hochgeladen werden. Für fortgeschrittene Benutzer steht jedoch die komplette Mächtigkeit des Internets zur Verfügung, wie beispielsweise PHP-Skripte oder Hooks für `git`.

Zu diesem Zeitpunkt ist die Benutzung dieses Systems freiwillig. Das Ziel ist es allerdings, aufs Herbsemester 2017 alle D-MATH-Vorlesungsseiten über dieses System laufen zu lassen. Dadurch sind die Offenheit der Daten, Archivierbarkeit und eine Konsistenz in der Darstellung garantiert.

2 Wie fange ich an?

2.1 Einrichten und Installation

Schreib Heinz eine E-Mail, dass du mitmachen willst! Wenn du ihm unter heinz.rasched@math.ethz.ch mitteilst, welche Vorlesung du betreust, erstellt er dir ein `git`-Repositorium und eine erste Version deiner Website, bei der schon vieles stimmen sollte.

Dann brauchst du `git`. Auf den D-MATH-Rechnern sollte dieses Tool bereits überall installiert sein. Falls du `git` selbst installieren musst, findest du die Software unter `https://git-scm.com/downloads`. Um herauszufinden, ob `git`

installiert ist, gibst du `git --version` in der Kommandozeile (siehe unten) ein. Wenn `git` installiert ist, dann wird die Versionsnummer zurückgegeben, andernfalls erhält man die Meldung `bash: git: command not found`.

2.2 Repositories lokal verwenden

Um `git` zu verwenden, benützt du die Kommandozeile in Linux bzw. Mac OS, die über das "Terminal" bzw. die Konsole bedient wird. Alternativ kann man sich auch graphische Interfaces zu `git` installieren. Solltest du nicht mit der Kommandozeile vertraut sein, dann finden sich im www unter dem Begriff "UNIX commands" Auflistungen der notwendigen Befehle.

Sobald das von Dir gewünschte Repositorium (fortan Masterrepositorium genannt) eingerichtet wurde, erhältst du von Heinz einen URL von der Form `https://metaphor.ethz.ch/XYZ/COURSEID`, der auf das Masterrepositorium verweist, wo sich die Ressourcen zur Website befinden. Du wirst die Website nicht direkt im Repositorium bearbeiten. Stattdessen erstellst du eine lokale Kopie dieses Repositoriums, wo du die Änderungen vornimmst, die du dann jeweils dem Masterrepositorium mitteilst. Hierfür wählst du einen Ordner `PATH`, wo du die Kopie erstellen willst und erstellst die gewünschte Kopie dort.

```
$ cd PATH
$ git clone https://metaphor.ethz.ch/XYZ/COURSEID
$ cd COURSEID
```

Mit dem letzten Befehl haben wir gleich in den lokal neu erstellten Ordner gewechselt, der eine Kopie des Inhalts des Repositorium enthält. Von nun an werden wir in dieser lokalen Kopie arbeiten und regelmässig unsere vorgenommenen Änderungen dem Hauptrepositorium mitteilen.

Heinz wird dir auch gleich die URL deiner Vorlesungshomepage mitteilen. Die erhaltene URL kannst du gleich in eDoz als Hauptlink für die Vorlesung hinterlegen, dann wissen auch alle Studierenden der Vorlesung, dass sie dort alle Infos finden.

3 Wie bearbeite ich meine Website?

3.1 Lokale Kopie bearbeiten

Durch `git clone` sind eine Menge Dateien – Beispiele für Übungsblätter, Skripten u.s.w. – in deinem aktuellen Verzeichnis `PATH/COURSEID` gelandet. Es befindet sich dort in der HTML-Datei `index.htm` eine lokale Kopie der Website, die bereits mit den aus dem eDoz und dem HA-Tool verfügbaren Inhalten versehen wurde, die aber mit den noch fehlenden Inhalten gefüllt werden sollte.

HTML ist eine Markup-Sprache wie LaTeX und dementsprechend enthält die Datei `index.htm` grundsätzlich alle Inhalte sowie Formatierungsangaben, dazu aber auch noch Verweise, die dem Browser sagen, wo er weitere Dateien findet, die er zur Darstellung benötigt. Ganz ähnlich wie die `documentclass` oder die Pakete in LaTeX eben.

Wie bei LaTeX kannst du die Datei `index.htm` in einem beliebigen Texteditor bearbeiten und deine Änderungen vornehmen. Die Struktur dieser Datei sollte einen Mathematiker nicht vor unüberwindbare Probleme stellen. Falls doch, löst man dieses Problem wie in der Mathematik und holt sich das passende Buch oder fragt jemanden, der schon weiss, wie es geht – letztere Variante ist vermutlich schneller.

Im Repositorium können, wie die Beispiele zeigen sollten, auch andere Dokumente zur Vorlesung abgelegt werden, auf die über die Website Zugriff gewährt werden soll.

Um den Effekt der vorgenommenen Änderungen zu begutachten, öffnest du die lokale Kopie der Website `index.htm` mit einem Browser.

3.2 Änderungen veröffentlichen

Alle Änderungen, die du vorhin vorgenommen hast, betreffen nur die lokale Kopie der Website `index.htm`. Um die Änderungen zu veröffentlichen, musst du nun `git` verwenden, um die lokal vorgenommenen Änderungen an das Repositorium zu senden, auf welches beim Besuch der Website zugegriffen wird.

Der Arbeitsfluss mit `git` sieht üblicherweise wie folgt aus:

1. Lokales Repositorium und Masterrepositorium werden lokal synchronisiert.
2. Ein Arbeitsschritt wird im lokalen Repositorium ausgeführt.
3. Die in der lokalen Kopie vorgenommenen Änderungen sowie neue Dateien werden dem Abbild des Repositoriums (dem sog. Index) hinzugefügt.
4. Der Zustand des Index wird "committed", d.h. der gegenwärtige Status des Index wird von `git` als neues Abbild des Repositoriums gespeichert.
5. Die Schritte 2-4 werden beliebig oft wiederholt.
6. Die vorgenommenen und committeden Änderungen werden dem Masterrepositorium mitgeteilt. Erst nach diesem Schritt werden die Änderungen online sichtbar.

Zur Ausführung der obigen Schritte wird wieder die Kommandozeile verwendet.

0. Du wechselst zuerst in den Ordner, wo die zu bearbeitenden Daten liegen:
   `$ cd PATH/COURSEID`
1. Synchronisiere die lokale Version mit dem Masterrepositorium:
   `$ git pull`
3. Du fügst die n geänderten/neuen Dateien/Verzeichnisse zum Index hinzu:
   `$ git add ./PATHTOFILE1 ./PATHTOFILE2 ... ./PATHTOFILEn`

4. Du committest als ganzes die Veränderungen am Index:
   `$ git commit`
   Nach Ausführung dieses Befehls wird ein Texteditor geöffnet, wo eine sog. "Commitmessage" festgehalten wird, die die vorgenommenen Änderungen beschreibt. Aus Gründen der Nachverfolgbarkeit und um die Versionierung ausnutzen zu können, lohnt es sich, relativ kleine Arbeitsschritte separat zu committen.

   Standardmässig wird der Editor `vi` aufgerufen, dessen Handhabung für Neulinge etwas gewöhnungsbedürftig ist. Ganz kurz: Die Tastenfolge

   `iTEXT<ESC>:wq<ENTER>`

   fügt `TEXT` ein und führt den Befehl aus. Hierbei sind mit <ESC> und <ENTER> das Drücken der Tasten "Escape" und "Enter" gemeint.

6. Das veränderte Abbild wird an das Masterrepositorium geschickt:
   `$ git push`
   Danach sind die committeden Änderungen dem Masterrepositorium hinzugefügt und werden auch bei Besuch der Website ersichtlich.

Nochmals kompakt:

```
$ git pull
$ git add ./PATHTOFILE1 ... ./PATHTOFILEn
$ git commit
$ git push
```

Ein Beispiel für den Arbeitsablauf findest Du am Ende dieses Dokuments.

Vorsicht: Alles, was du so hochlädst, kann prinzipiell von jedem eingesehen werden, der die Adresse kennt!

3.3 Bemerkungen

Häufig wird man mehrere Arbeitsschritte ausführen und erst am Ende "committen". In dem Fall kann man die Schritte 3 und 4 am Ende ausführen, d.h. die veränderten Dateien aus einem logischen Arbeitsschritt zusammen zum Index hinzufügen, danach den neuen Zustand mittels commit speichern und dann dasselbe für den nächsten logischen Arbeitsschritt wiederholen.

Wenn Ihr **zusammen** in einem Repositorium arbeitet, vergesst nicht, Schritt 1 regelmässig auszuführen, um Konflikte auf ein Minimum zu reduzieren. Normalerweise lassen sich Konflikte mit `git` relativ leicht beheben (Änderungen zusammenführen/"mergen"), aber es ist dennoch einfacher, dieses Problem zu umgehen, indem man immer nur die aktuellste Version verändert.

`git` ist eine Versionierungssoftware und kann bedeutend mehr als das was oben aufgelistet ist. Eine Version des Manuals findet sich unter `https://metaphor.ethz.ch/x/man/git/`. Dort werden auch nützliche Begriffe wie `git revert, git rm, git mv` erklärt. Letztere werden verwendet um Dateien zu entfernen oder zu verschieben. Ein besonders nuetzlicher Befehl ist `git status`, der die veränderten Dateien auflistet.

4 Der `src`-Ordner

Wer Interesse hat, kann zudem bei Heinz ein sog. source-Repositorium beantragen. Dieses dient der Ablage und Bearbeitung von Dokumenten und Daten zur Vorlesung. Beispielsweise lassen sich dort die Übungsserien, Skripten sowie Prüfungen erstellen und ablegen, so dass diese Daten einerseits mit Versionskontrolle und andererseits auf Servern der ETH abgespeichert sind. Zudem besteht so die Möglichkeit, dass die MC-Fragen und Korrekturen von Nemesis direkt dort herausgelesen werden, so dass kein Emailverkehr notwendig ist.

5 Weiteres

Falls du wissen willst, wie Heinz Rasched deine HTML-Seite automatisch mit Daten gefüllt hat und du gerne programmierst, darfst du gerne mit ihm Kontakt aufnehmen und weitere Automatisierungen ins Auge fassen. Dies kommt allen aktuellen und künftigen Übungsorganisatoren zu Gute!

Allgemeine Kritik und Anregungen gehen bitte an Andreas Steiger: andreas.steiger@math.ethz.ch

6 Ein Beispiel

Im Folgenden fügen wir Serie 15 zur Website zur Vorlesung 401-1151-00L hinzu.

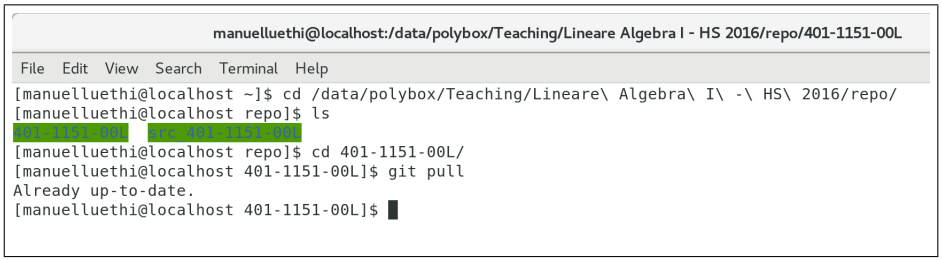


Abbildung 1: Gehe zur lokalen Kopie des Repositoriums und synchronisiere.

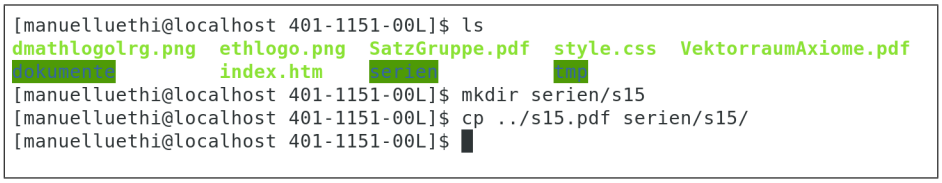
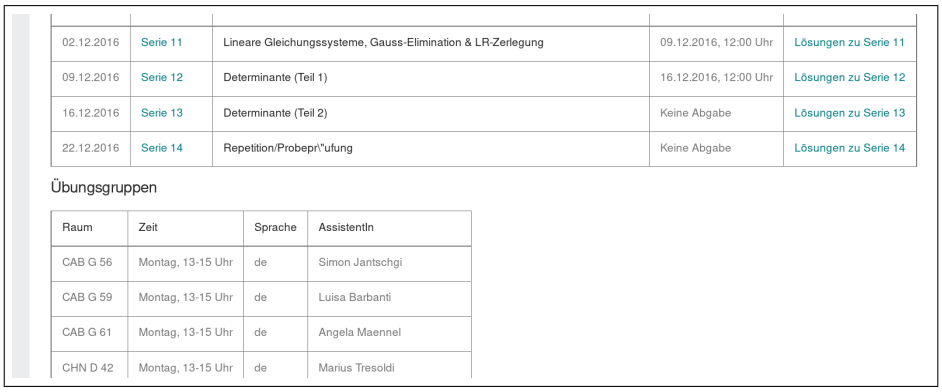


Abbildung 2: Speichere lokal die Serie 15 am richtigen Ort ab (hier befand sich das Original in `./../../s15.pdf`, aber das ist natürlich vom Computer abhängig).



02.12.2016	Serie 11	Lineare Gleichungssysteme, Gauss-Elimination & LR-Zerlegung	09.12.2016, 12:00 Uhr	Lösungen zu Serie 11
09.12.2016	Serie 12	Determinante (Teil 1)	16.12.2016, 12:00 Uhr	Lösungen zu Serie 12
16.12.2016	Serie 13	Determinante (Teil 2)	Keine Abgabe	Lösungen zu Serie 13
22.12.2016	Serie 14	Repetition/Probepr\"ufung	Keine Abgabe	Lösungen zu Serie 14

Übungsgruppen

Raum	Zeit	Sprache	AssistentIn
CAB G 56	Montag, 13-15 Uhr	de	Simon Jantschgi
CAB G 59	Montag, 13-15 Uhr	de	Luisa Barbanti
CAB G 61	Montag, 13-15 Uhr	de	Angela Maennel
CHN D 42	Montag, 13-15 Uhr	de	Marius Tresoldi

Abbildung 3: Die Website vor Anpassung des Inhalts.

```
      <td align=left>Keine Abgabe</td>
      <td align=left><a href="serien/s13/l13.pdf">L&ouml;sungen zu Serie 13</a></td>
    </tr>
    <tr>
      <td align=left>22.12.2016</td>
      <td align=left><a href="serien/s14/s14.pdf">Serie 14</a></td>
      <th align=left>Repetition/Probepr\"ufung</th>
      <td align=left>Keine Abgabe</td>
      <td align=left><a href="serien/s14/l14.pdf">L&ouml;sungen zu Serie 14</a></td>
    </tr>
  </table>
  <h2>&Uuml;bungsgruppen</h2>
  <table class="table1">
```

Abbildung 4: Öffne `index.htm` im gewünschten Texteditor und gehe zur Stelle, die verändert werden soll.

```
      <td align=left>Keine Abgabe</td>
      <td align=left><a href="serien/s13/l13.pdf">L&ouml;sungen zu Serie 13</a></td>
    </tr>
    <tr>
      <td align=left>22.12.2016</td>
      <td align=left><a href="serien/s14/s14.pdf">Serie 14</a></td>
      <th align=left>Repetition/Probepr&uuml;fung</th>
      <td align=left>Keine Abgabe</td>
      <td align=left><a href="serien/s14/l14.pdf">L&ouml;sungen zu Serie 14</a></td>
    </tr>
    <tr>
      <td align=left>03.01.2017</td>
      <td align=left><a href="serien/s15/s15.pdf">Serie 15</a></td>
      <th align=left>Test</th>
      <td align=left>Keine Abgabe</td>
      <td align=left>Link zu den L&ouml;</td>
    </tr>
  </table>
```

Abbildung 5: Die bearbeitete Stelle: ein Eintrag zu Serie 15 wurde erstellt und ein Typo korrigiert.

```
[manuelluethi@localhost 401-1151-00L]$ git status
On branch master
Your branch is up-to-date with 'origin/master'.
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   index.htm

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        serien/s15/

no changes added to commit (use "git add" and/or "git commit -a")
[manuelluethi@localhost 401-1151-00L]$
```

Abbildung 6: Mit `git status` überprüfst Du, welche veränderten Dateien sich im lokalen Ordner befinden, die nicht zum Index hinzugefügt worden sind.

```
[manuelluethi@localhost 401-1151-00L]$ git status
On branch master
Your branch is up-to-date with 'origin/master'.
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   index.htm

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        serien/s15/

no changes added to commit (use "git add" and/or "git commit -a")
[manuelluethi@localhost 401-1151-00L]$ git add serien/s15/*
[manuelluethi@localhost 401-1151-00L]$ git commit
```

Abbildung 7: Mit `git add serien/s15/*` fügen wir den Ordner `serien/s15` und seinen gesamten Inhalt zum Index hinzu. Mit `git commit` speichern wir die aktuelle Version des Index ab.

```
added sheet 15 to the repo
# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
# On branch master
# Your branch is up-to-date with 'origin/master'.
#
# Changes to be committed:
#       new file:   serien/s15/s15.pdf
#
# Changes not staged for commit:
#       modified:   index.htm
#
~
~
~
```

Abbildung 8: Das Terminal nach Ausführung von `git commit`: Je nach Einstellung öffnet sich ein anderer Texteditor (hier `vi`), in welchem wir die commit-message festhalten.

```
[manuelluethi@localhost 401-1151-00L]$ git commit
[master 3fce550] added sheet 15 to the repo:x added sheet 15 to the repo
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 serien/s15/s15.pdf
[manuelluethi@localhost 401-1151-00L]$ git add index.htm
[manuelluethi@localhost 401-1151-00L]$ git commit
```

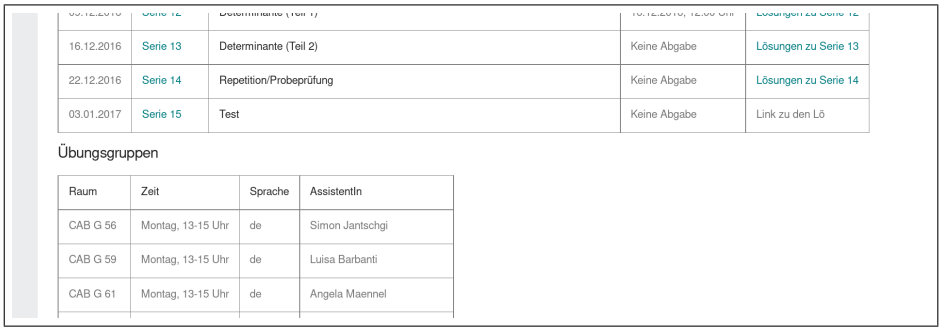
Abbildung 9: Im nächsten Schritt fügen wir die Veränderung in `index.htm` zum Index hinzu: Die neu verfügbare Serie 15 wurde verlinkt.

```
added link to sheet 15 to index.htm (and corrected type)
# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
# On branch master
# Your branch is ahead of 'origin/master' by 1 commit.
#   (use "git push" to publish your local commits)
#
# Changes to be committed:
#       modified:   index.htm
#
~
~
~
~
~
```

Abbildung 10: Wieder verfassen wir eine entsprechende commit-Nachricht (mit Typo in typo).

```
[manuelluethi@localhost 401-1151-00L]$ git push
Counting objects: 7, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (6/6), done.
Writing objects: 100% (7/7), 805 bytes | 0 bytes/s, done.
Total 7 (delta 4), reused 0 (delta 0)
remote: Already on 'master'
To https://metaphor.ethz.ch/git/lhm/2016/hs/401-1151-00L
   670617b..b59c411  master -> master
[manuelluethi@localhost 401-1151-00L]$
```

Abbildung 11: Wir geben die Änderungen mit `git push` an das Masterrepositorium weiter.



16.12.2016	Serie 13	Determinante (Teil 2)	Keine Abgabe	Lösungen zu Serie 13
22.12.2016	Serie 14	Repetition/Probeprüfung	Keine Abgabe	Lösungen zu Serie 14
03.01.2017	Serie 15	Test	Keine Abgabe	Link zu den Lö

Übungsgruppen

Raum	Zeit	Sprache	AssistentIn
CAB G 56	Montag, 13-15 Uhr	de	Simon Jantschgi
CAB G 59	Montag, 13-15 Uhr	de	Luisa Barbanti
CAB G 61	Montag, 13-15 Uhr	de	Angela Maennel

Abbildung 12: Der entsprechende Ausschnitt der Website nach dem `git push` Befehl.