Class 8: Semester Org Files (3/4): Lectures

Table of Contents / Agenda

1 The `Lectures` Top-level Tree **1**

2 Individual Lecture Subtree **3**

1 The `Lectures` Top-level Tree

- The `Lectures` top-level tree, which is the basis for the class schedule table, contains a subtree for each class and event (e.g., assignment, exam). See the example below.

```
* Lectures
:PROPERTIES:

#+NAME: Update Classes
#+BEGIN_SRC emacs-lisp :results none

** Tasks [0/1]                                          :noexport:skipcount:
** Classes and Assignments Dates                                :skipcount:
** Introduction
** Org Mode Basics
** ...
** _Assignment 1 Due_                    :Assignment:skipcount:
** _Exam 1_                                                        :Exam:
** *Thanksgiving Holiday*                              :Holiday:skipcount:
** _Final Exam_                                                 :skipcount:
```

1.1 `:PROPERTIES:`

- The properties of the `Lectures` top-level tree contain information common across lectures such as export options, including the handout LaTeX headers (e.g., syntax highlighting options for the `minted` package). With the `org-use-property-inheritance` option set to `t`, the settings will be propagated to its subtrees. For example, the `:EXPORT_LATEX_HEADER+:` property items, which specify the preamble for LaTeX lecture handouts, will be shared by all lecture subtrees.

- The `COLUMNS` and `ID` properties of the `Lectures` top-level tree are used to create the class schedule columnview dynamic block described in `Syllabus`.

```
:PROPERTIES:
:COLUMNS: %Date %Class %ITEM
:ID:  79d5e887-4637-43e7-8e8a-b83fa83ee56e
...
:END:
```

1.2 `Update lecture metadata` Source Code Block

- The `Lectures` top-level tree has a source code block named `Update Lectures`. When executed with `C-c C-c`, it is designed to go through each lecture subtree and perform the following actions:

  1. Update `:PROPERTIES:` for the lecture, such as the class number (`:CLASS:`), class date (`:DATE:`), and file name of the export (`:EXPORT_FILE_NAME:`).

     `:CLASS:` The lectures are assumed to be in the order of the lecture schedule (e.g., first lecture on Class 1). Note that any subtree with a `:skipcount:` tag will be ignored, which is useful for non-lecture subtrees (e.g., assignment due dates and holidays).

     `:DATE:` It will get the date of each class from the `DATE_CLASS_XX` file-level properties, which are defined in the `Lectures and Assignments Dates` subtree.

     `:EXPORT_FILE_NAME:` By default, the script sets the `:EXPORT_FILE_NAME:` as the subtree heading, which can be overridden by setting the `:EXPORT_FILE_NAME_MANUAL:` property of the lecture subtree. If the property exists, the script will use its value for `:EXPORT_FILE_NAME:` instead. This is useful when the lecture subtree heading is very long or contains invalid characters for a file name.

  2. Update `Lecture Agenda` under the `Introduction` subtree.

     `Lecture Agenda` The script will get the list of subtrees that belong to the particular lecture, ignoring any with `noexport`, `handoutonly`, or `slideonly` tags. Then, it will insert the list into the body of `Lecture Agenda`. In addition, it will set the `CUSTOM_ID` property value of the subtree accordingly, so the agenda can be used in other places.

  3. Update `Last Class` under the `Introduction` subtree and `Class Summary` of each lecture.

     `Last Class` The script will insert an `#+INCLUDE:` statement which points to the previous lecture's `Lecture Agenda` subtree under the `Introduction`. This is to provide a recap of the previous lecture prior to starting the current lecture.

     `Class Summary` The script will insert an `#+INCLUDE:` statement which points to the current lecture's `Lecture Agenda` subtree under the `Introduction`. This provides a summary of the current lecture.

- The user should run this script before updating the class schedule table in the `Syllabus`, so the most current information is reflected in the table.

1.3 `Lectures and Assignments Dates` Subtree

- In this subtree, instructors can define lecture dates and assignment due dates as file-level properties. For example,

```
#+MACRO: DUE_ASSIGNMENT_1 [2018-09-27 Thu]
#+MACRO: DUE_ASSIGNMENT_2 [2018-10-30 Tue]

#+DATE_CLASS_01: [2018-08-28 Tue]
#+DATE_CLASS_02: [2018-08-30 Thu]
```

- The `Update lecture metadata` source code block will use the dates defined in the file-level properties as shown above to update the date of each lecture.
- 28 lecture/class dates are pre-defined in the Org-Coursepack. Instrutors can easily customize them to meet their needs.

1.4 `Common Items` Subtree

- The `Common Items` subtree has common items across all lectures. Currently there is one subtree, `Handout heading`, which contains LaTeX codes for header items and table of contents. The content will be included from the `Handout heading` subtree of each individual lecture subtree.

1.5 Dynamic Columnview of Lectures

- A useful functionality of Org mode is the ability to create a table-view overlay of subtrees with their property values. Instructors can use the `org-columns` command to create a column-view of lectures, which is essentially the same as the class schedule table in the `Syllabus`. It is useful when there is a need to quickly inspect the overall course schedule.

2 Individual Lecture Subtree

Each lecture subtree contains the teaching materials for that particular lecture/class. The example below shows the general structure of the subtree.

```
** Course Introduction
:PROPERTIES:
*** Tasks [0/1]                                                  :noexport:
*** Handout heading                                 :handoutonly:ignore:
*** Introduction and Recap of Last Class                        :slideonly:
*** Introduction to {{{COURSE}}}
*** Overview of the Directory Structure
*** Class Recap                                                 :slideonly:
```

2.1 :PROPERTIES:

- A lecture subtree has properties containing lecture-specific information.
- As described earlier, :CLASS: (class number), :EXPORT_FILE_NAME:, and :DATE: (class date) will be automatically updated by the Update lecture metadata source code block.
- The :EXPORT_TO: property has clickable links written in Emacs-lisp, which will export class content to the designated output format. For example, clicking reveal.js will export content to reveal.js slides.
- The :OUTPUT_VIEW: property has links that, when clicked, opens the corresponding output files, such as html or pdf files. The links will use the value of the :EXPORT_FILE_NAME: property as the file path; hence, it is unnecessary to manually edit the output links.

```
** Course Introduction
:PROPERTIES:
:CLASS:    1
:EXPORT_TITLE: Class {{{property(CLASS)}}}: {{{property(ITEM)}}}
:EXPORT_FILE_NAME: ./Lectures/01 Course Introduction
:DATE:     [2018-08-28 Tue]
:EXPORT_TO:  reveal.js | Beamer | LaTeX
:OUTPUT_VIEW: HTML | PDF
:END:
```

2.2 Tasks

The Tasks subtree contains lecture-specific tasks you may have as an instructor. These are presented in the form of Org mode TODO items. The :noexport: tag prevents the tree from being exported.

2.3 Handout heading

The Handout heading headline will only be included in a handout export (with the :handoutonly: tag). It includes the content from Handout heading headline of the Common Items subtree in the Lectures top-level tree.

2.4 Introduction and Recap of Last Class

Classes often begin with a recap of the previous lecture topics and a preview of the current lecture topics. The Introduction and Recap of Last Class has three subheadings:

```
*** Introduction                                                  :slideonly:
**** Announcements
**** Recap of Last Class
**** Lecture Outline
```

Instructors can enter any announcements to be made in class in `Annoucements`; `Recap of Last Class` includes a recap of the learning objectives from the previous class; `Lecture Outline` lists the learning objectives for the current lecture.

Note that the content (`#+INCLUDE:` statements) and properties (e.g., `CUSTOM_ID`) of the latter two subheadings will be automatically updated by the `Update lecture metadata` script as discussed earlier. Hence, users do not need to manually edit these.

With the `:slideonly:` tag, the `Introduction and Recap of Last Class` headline will only be exported to slide outputs.

2.5 Content

Subtrees following the `Introduction and Recap of Last Class` subtree contain lecture content. To minimize redunancy, lectures should draw as much material from the reusable content in the topic Org file subtrees as possible. See example below. (For more examples, see `2020 Spring.org` in `Org_Teaching`.)

```
*** Topic Org Files
#+INCLUDE: "../../../Topics/Org_Teaching.org::#Lectures/Topic Org Files" :only-contents t
*** Course Org Files
#+INCLUDE: "../../../Topics/Org_Teaching.org::#Lectures/Course Org Files" :only-contents t
```

2.6 Class Recap

The `Class Recap` reviews the content of the current class, by including the content of `Lecture Outline` in the `Introduction and Recap of Last Class` subtree. The `#+INCLUDE:` statement will be automatically generated by the `Update lecture metadata` script as described earlier. With the `:slideonly:` tag, `Class Recap` will only be exported in slide outputs.