Moodle 4.0 course creation

Moodle Moot It 2021
02 12 2021
The long road to 4.0

- August 2020: Moodle 4.0 as a 18 month release is announced
- March 2021: the first 4.0 commit was integrated
- October 2021: code freeze for Moodle 4.0
- ~January 2022: estimated release date of Moodle 4.0
Moodle HQ 4.0 projects

- Moodle HQ projects for Moodle 4.0 are divided in 3 main projects:
  - Navigation
  - Course creation (this session is about this project 😊)
  - Student experience
- Plus the integration of the report builder thanks to the Moodle Workplace Team (next session 😊)
PART I

COURSE CREATION NEW FEATURES

- Course navigation
- Activities card design
- Collapsible sections
- Course index
- Blocks drawer
- Drag & Drop
- Add sections
- Move modal
- Completion icons
- Accessibility
Demo time!

Because nothing can go wrong with an old-fashioned live demo
PART 2

MIGRATION PROCESS

- Courseformat subsystem
- Output classes
- Format renderer
- Base class
- HTML data attributes
Demo time!

What happens when you use a Moodle 3.x course format in Moodle 4.0
3.11 formats will use the previous course editor until 4.3

However, they won’t have some of the 4.0 new features:

- Course index
- New drag and drop
- Move modal
- Add sections without reloading
Enabling debug on a 3.11 format will guide you through all the deprecated methods.

Take in mind that those methods will be removed in Moodle 4.3
The course/format folder is now a subsystem containing:
- course format renderings and rendering methods
- course tests
- course format API

Related issues: MDL-71863 and MDL-72578.

Important: all deprecated renderer methods using html_writer will be removed for good in Moodle 4.3
Course format output classes

- Added **3 main renderer methods** (for full course, section and activity)
- The format **renderer class is now mandatory**
- 17 new output classes + mustache templates
- 33 deprecated methods from course_renderer and section_renderer (aka. format_section_renderer_base in Moodle 3.11)
- 13 new methods in core_courseformatase class (aka. format_base in Moodle 3.11)
Don’t panic. I will explain this diagram right now.
4.0 new features

Course index: Opt-in

Indentation: Opt-out
(disabled in the standard formats)

New drag&drop: Opt-in

Move section/activity modal: Opt-in

Collapsible sections: Opt-out
(overridable)

Activity card display: Opt-out
(overridable)
4.0 Format base class methods

- `uses_sections`
  - Course sections

- `uses_course_index`
  - Course index

- `uses_indentation`
  - Activity indentation

- `supports_ajax`
  - Ajax edition (mandatory)
  - Move activity modal
  - New Drag & Drop
  - Ajax add section
  - Collapsible sections

- `supports_components`
  - Replace `PAGE->user_is_editing`
  - Replace `PAGE->get_renderer`

- `show_editor`

- `get_renderer`

- `get_section_number`

- `set_section_number`

- `get_sections_preferences`

- `delete_format_data`

- `get_section_highlighted_name`

- `page_title`
  - Collapsible sections
  - Highlighted label text
  - Customise course page title
Course HTML structure

The new course UI components use **data attributes to identify elements** (instead of CSS classes and tag names):

- **Sections**: data-for="section" data-id="XXX" data-number="XXXX"
  - **Section title**: data-for="section_title" data-id="XXX" data-number="XXXX"
  - **Activity**: data-for="cm" data-id="XXX"

The previous CSS classes are still used for some legacy actions but it will be deprecated for good in Moodle 4.3
PART 3

COURSE EDITOR ARCHITECTURE

- Reactive state
- State actions
- Components
- Mutations
Demo time!

“Pay no attention to that man behind the curtain”
- L. Frank Baum (The Wizard of Oz)
Reactive state principles

1. Each reactive element is controlled by its own component class
2. The UI is and updated representation of a state data
3. All user interactions can alter the state data but never the HTML
4. The new services are designed to alter the current state data, not the interface.
Module: core/reactive

Provides several classes to create small reactive states in Moodle.

- **Reactive**: each reactive application is represented by an instance of this class.
- **BaseComponent**: all reactive components extends this class
- **DragDrop**: a class to convert components into draggable and/or dropzones.
- **M.cfg.reactive**: object to debug your reactive application from the javascript console
Reactive workflow

- User Interface
  - User Interactions
  - Ask for state changes
  - Notify state changes to watchers

- Components
  - UI update

- Reactive instance
  - Access to the state data
  - Dispatch state mutations
  - Mutate the state

- State Manager
  - Mutate the state

- State
  - Call Web services

- Mutations library

Each reactive APP responsibility:
- Must extend and implement the logic
- Just extend with no major changes
- Use it as it is
Thanks for your attention!