



The Odoo JS Framework

A large, semi-transparent watermark of the word "odoo" in the same lowercase, rounded font as the logo, centered on the slide.

Gery Debongnie (GED) - RD Framework Team

- 1 Introduction
- 2 Overview: JS Framework
- 3 New Views
- 4 New Testing Framework
- 5 Documentation/Guidelines
- 6 The Future

A large, white, sans-serif number '1' is centered within a thick, light-gray circular outline.

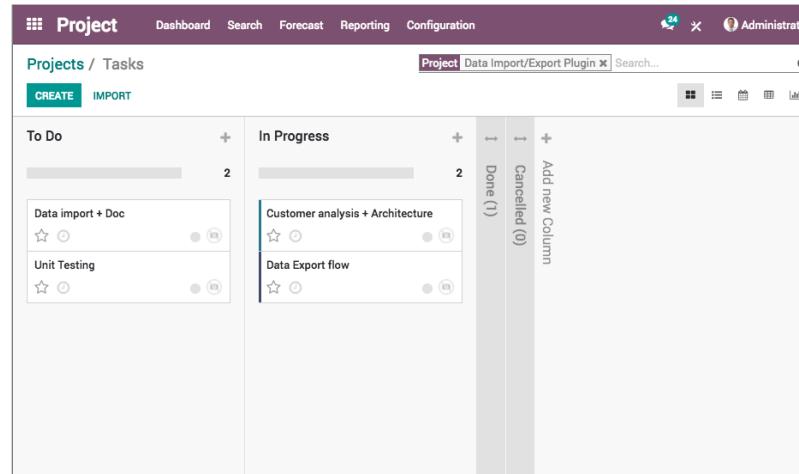
Introduction

What's new in v11?

Odoo JS in 2017:

Odoo JS in 2017:

We rewrote the javascript views...



The screenshot shows a Kanban view within the Odoo JS framework. The interface includes a top navigation bar with links for Dashboard, Search, Forecast, Reporting, and Configuration, along with a user icon for Administrator. A search bar at the top right contains the text "Project Data Import/Export Plugin". The main area displays four columns of tasks:

- To Do**: Contains two items: "Data import + Doc" and "Unit Testing".
- In Progress**: Contains two items: "Customer analysis + Architecture" and "Data Export flow".
- Done**: Contains one item: "Cancelled (1)".
- Cancelled**: Contains one item: "Cancelled (1)".

Each task card includes a star rating icon, a circular progress bar, and a small trash bin icon. A "CREATE" button is located at the top left of the view, and a "Import" link is visible above the "To Do" column. A "Search..." input field is also present at the top right.

“ Using SLOC to measure software progress is like using kg for measuring progress on aircraft manufacturing ”

Odoo v11 in a nutshell (JS)

	v10	v11
LOC	33.8k	32.8k
Comments	3.0k	12.5k
Tests LOC	3.9k	29.8k
Assets size (gzipped)	1042kb	909kb

Also New in v11: JS unit tests

Web Tests

Hide passed tests Check for Globals No try-catch Filter: Go

QUnit 2.2.1; Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_6) Chrome/61.0.3163.91 Safari/537.36 sort by time (desc)

870 tests completed in 137738 milliseconds, with 0 failed, 1 skipped, and 0 todo.
4225 assertions of 4225 passed, 0 failed.

1. core > Registry: key set (1) Rerun	10 ms
2. core > Registry: extension (2) Rerun	0 ms
3. core > Registry: remain-linked (2) Rerun	0 ms
4. core > Registry: multiget (1) Rerun	0 ms
5. core > Registry: extended-multiget (1) Rerun	1 ms
6. core > PY.eval: simple python expression (2) Rerun	2 ms
7. core > PY.eval: not prefix (3) Rerun	1 ms
8. core > PY.eval: strftime (3) Rerun	3 ms
9. core > PY.eval: context_today (1) Rerun	1 ms
10. core > PY.eval: timedelta.test_constructor (16) Rerun	6 ms
11. core > PY.eval: timedelta.test_computations (28) Rerun	11 ms
12. core > PY.eval: timedelta.test_basic_attributes (3) Rerun	1 ms

much testing!



New in v11: various

- ▶ rpc: one method to rule them all
- ▶ lazy loading support
- ▶ class for manipulating domains/context
- ▶ better code organization, primitives for concurrency programming, ...
- ▶ deprecate dataset, Model

```
return this._rpc({  
    model: 'res.partner',  
    method: 'create',  
    args: [data],  
    context: context,  
});
```

and also

- ▶ more reusable widgets
- ▶ simpler code
- ▶ better communication between components
- ▶ more documentation
- ▶ coding guidelines
- ▶ ...

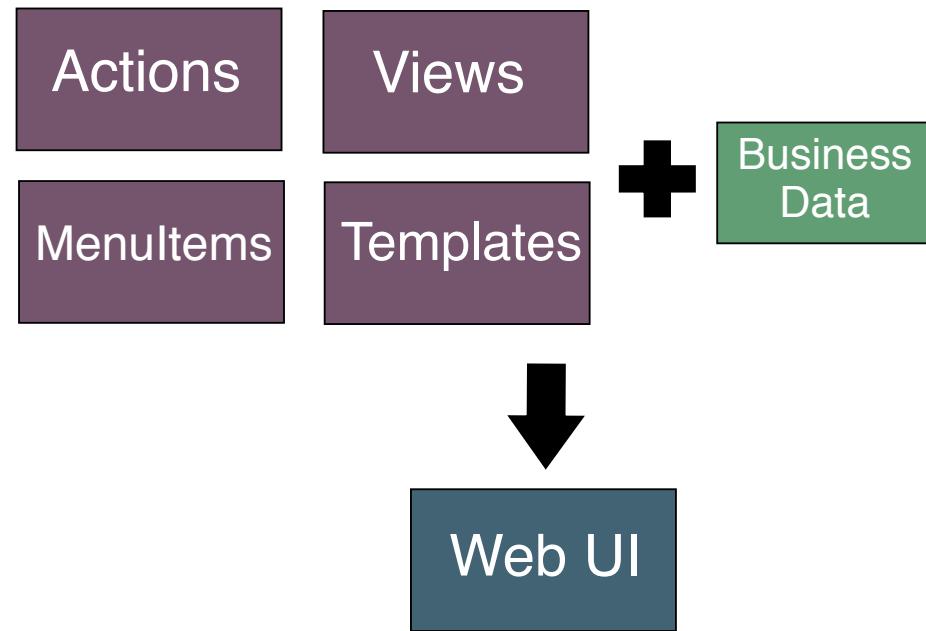
A large, bold number '2' is centered within a light gray circle. The circle has a thin black outline and a slight shadow, giving it a three-dimensional appearance.

2

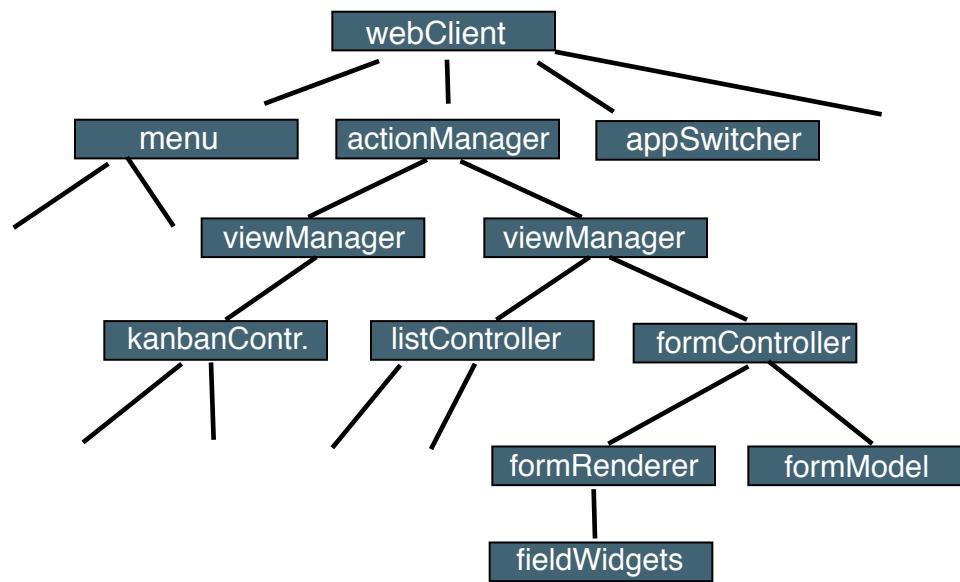
Odoo JS Framework

Yes, there's method in the madness!

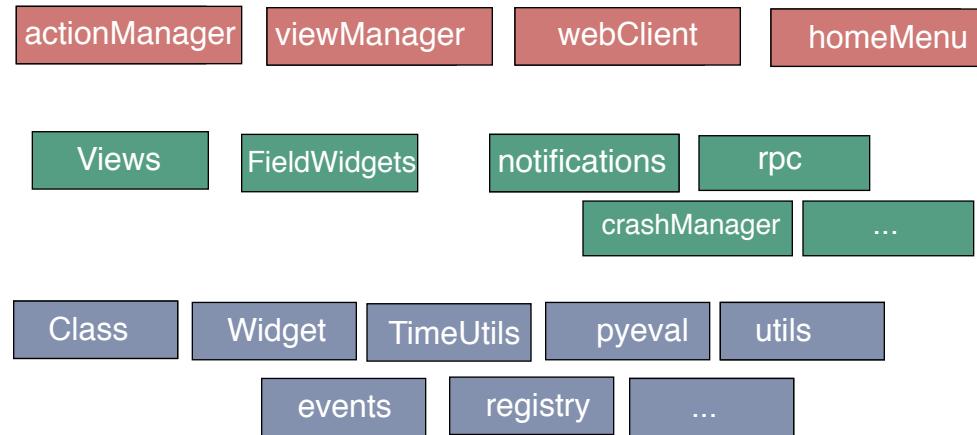
Web Client Architecture



Component Tree



Web Client Layers



Component Communication

Before:

main bus + reference hunting

```
var view_id = this.__parent.view.action_manager  
    .current_action.controller.view_id;
```

Component Communication

Before:

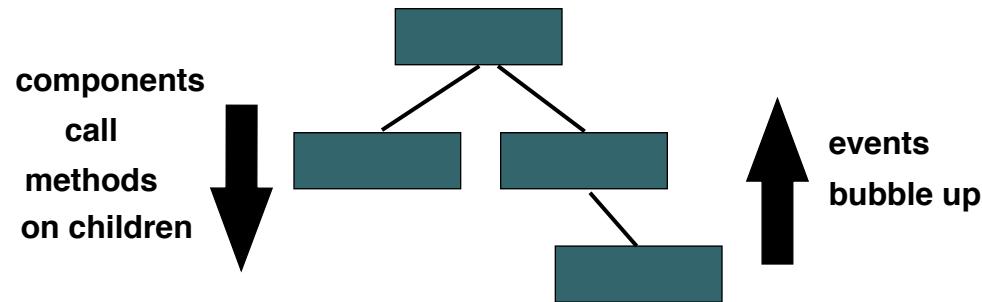
main bus + reference hunting

```
var view_id = this.__parent.view.action_manager  
    .current_action.controller.view_id;
```

Now and in the future:

- ▶ try to instantiate component with what it needs
- ▶ never fish for something in parent
- ▶ moving toward event bubbling up to communicate with environment

Communication



```
this.trigger_up('display_notification', {  
    title: 'Move Zig',  
    message: 'All your bases are belong to us',  
});
```

3

New JS Views

“ No, I will not allow you to rewrite the web client from scratch just for fun... ”

my manager
performance appraisal, 2016

New JS views

Full rewrite of JS views + 30k loc of unit tests

- ▶ Largest (framework) project in 2017

Thank you Team!!!

New JS views

Full rewrite of JS views + 30k loc of unit tests

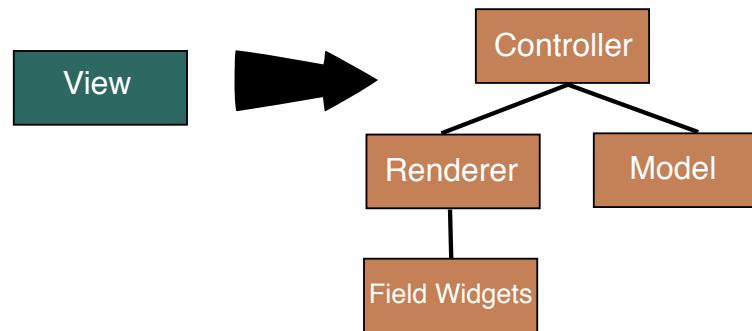
- ▶ Largest (framework) project in 2017

Thank you Team!!!

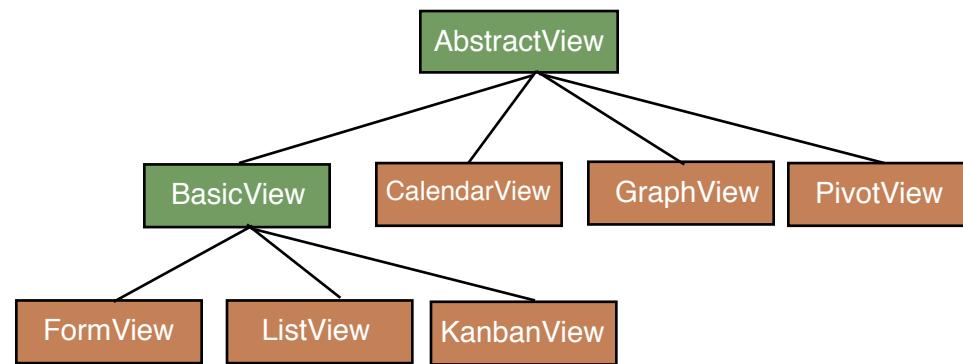
Motivations: reduce technical debt

- ▶ separate rendering from behaviour/model
- ▶ use views in other places
- ▶ use field widgets elsewhere
- ▶ unify behaviour across views
- ▶ ...

New Views: Overall Design



Class Hierarchy



Similar hierarchy for Controllers/Renderers/Models

BasicModel: Our Browser DB

- ▶ used by List, Form and Kanban
- ▶ about 3.7k LOC

it handles data fetching, onchanges, groups of records, relational fields, sorting, safepoints, rollbacks, modifiers, x2many command generation, ...

BasicModel: Our Browser DB

- ▶ used by List, Form and Kanban
- ▶ about 3.7k LOC

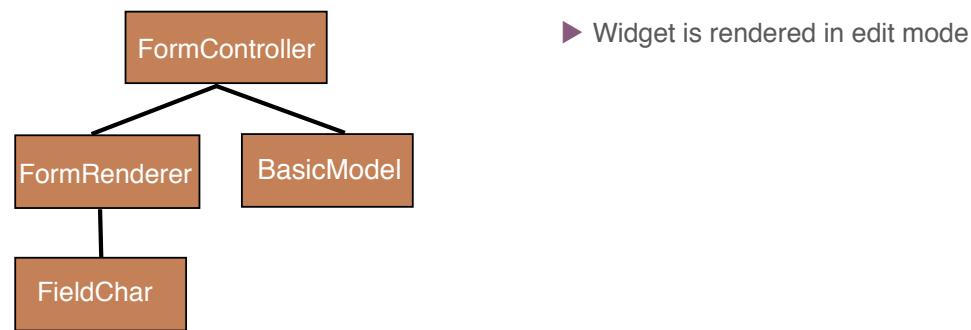
it handles data fetching, onchanges, groups of records, relational fields, sorting, safepoints, rollbacks, modifiers, x2many command generation, ...

[Here is a demo](#)

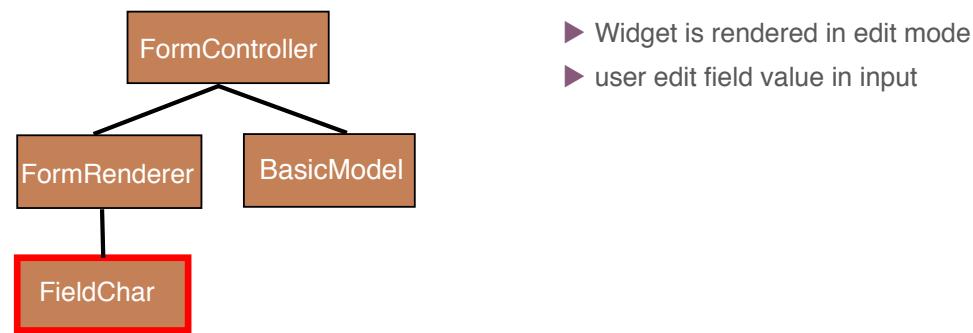
GED + BasicModel =



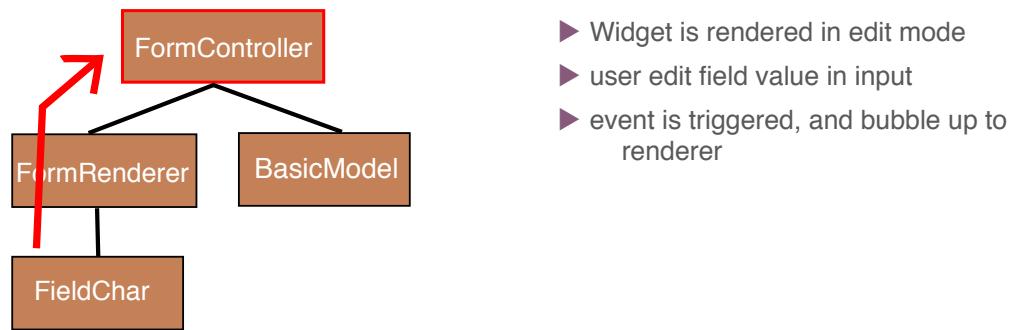
Field Widgets: change example



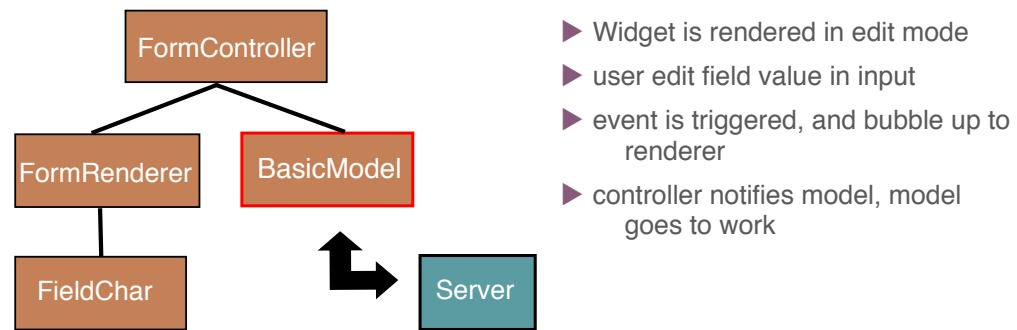
Field Widgets: change example



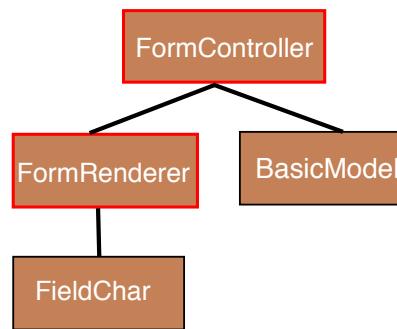
Field Widgets: change example



Field Widgets: change example

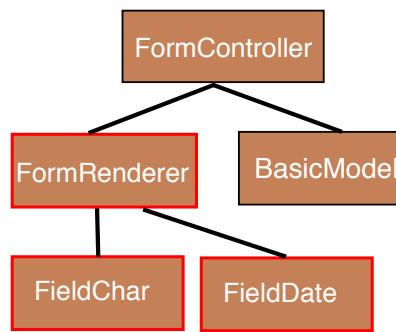


Field Widgets: change example



- ▶ Widget is rendered in edit mode
- ▶ user edit field value in input
- ▶ event is triggered, and bubble up to renderer
- ▶ controller notifies model, model goes to work
- ▶ controller confirms changes to renderer

Field Widgets: change example



- ▶ Widget is rendered in edit mode
- ▶ user edit field value in input
- ▶ event is triggered, and bubble up to renderer
- ▶ controller notifies model, model goes to work
- ▶ controller confirm changes to renderer
- ▶ renderer updates (changed) fields

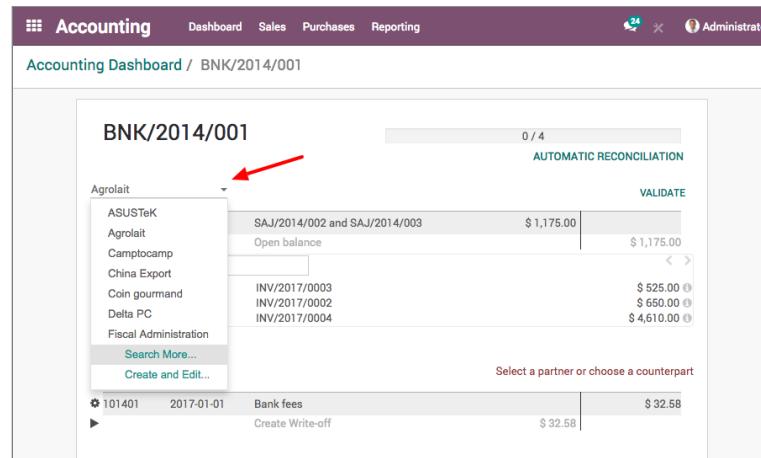
Just because we can doesn't mean we should

The screenshot shows a Kanban board with three columns: To Do, In Progress, and Done. The To Do column has 6 tasks, the In Progress column has 13 tasks, and the Done column has 5 tasks. Each task card contains a summary, due date, and a list of recent activities from 'YourCompany, Administrator'.

- To Do:** Document history management (September 22, 2017)
 - YourCompany, Administrator - 3 days ago: Dear customer, Thank you for your enquiry. If you have any questions, please let us know. Best regards,
 - YourCompany, Administrator - 3 days ago: Task opened
 - Project Manager: Demo User
 - Kanban State: In Progress
 - Project: Research & Development
 - Task Title: Document history management
 - Stage: To Do
 - Assigned to: Demo User
- In Progress:** Integrate Modules (September 22, 2017)
 - YourCompany, Administrator - 3 days ago: Task opened
 - Project Manager: Demo User
 - Kanban State: In Progress
 - Project: Website for Sales & WMS
 - Task Title: Integrate Modules
 - Stage: In Progress
 - Assigned to: Administrator
 - YourCompany, Administrator - 3 days ago: Internal testing + Software Install
 - Project Manager: Demo User
 - Kanban State: In Progress
 - Project: E-Learning Integration
 - Task Title: Internal testing + Software Install
 - Stage: In Progress
 - Assigned to: Administrator
- Done:** User Interface design (September 22, 2017)
 - YourCompany, Administrator - 3 days ago: Task opened
 - Project Manager: Demo User
 - Kanban State: In Progress
 - Project: E-Learning Integration
 - Task Title: User Interface design
 - Stage: Done
 - Assigned to: Administrator
 - YourCompany, Administrator - 3 days ago: Deploy and review on live system
 - Project Manager: Demo User
 - Kanban State: In Progress
 - Project: Website for Sales & WMS
 - Task Title: Deploy and review on live system
 - Stage: Done
 - Assigned to: Administrator

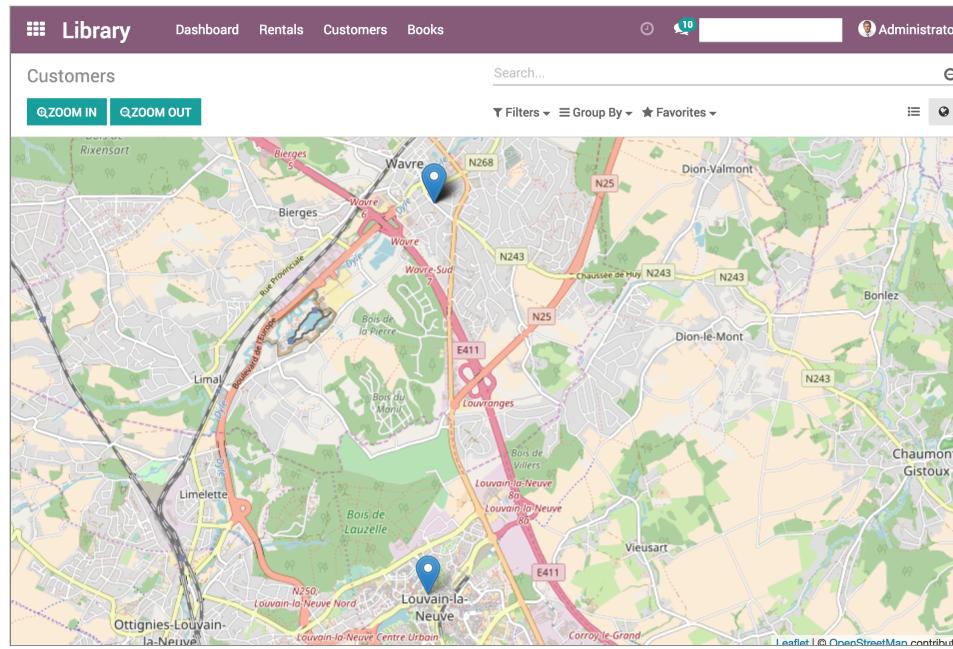
Views/Fields in the wild

- ▶ Views can now be instantiated anywhere
- ▶ same for fields



The screenshot shows the Odoo Accounting module's reconciliation interface. At the top, there's a navigation bar with tabs for Accounting, Dashboard, Sales, Purchases, and Reporting. The user is logged in as 'Administrator'. Below the navigation is a breadcrumb trail: Accounting Dashboard / BNK/2014/001. The main area displays a reconciliation window titled 'BNK/2014/001'. It shows two partners: 'Agrolait' and 'ASUSTeK'. A red arrow points to a dropdown menu next to 'Agrolait' which lists other partners: Agrolait, Campocamp, China Export, Coin gourmet, Delta PC, and Fiscal Administration. Below the dropdown is a 'Search More...' button. To the right of the dropdown is a 'VALIDATE' button. The reconciliation table has four columns: Date, Description, Amount, and Counterpart. The first row shows an open balance from 'SAJ/2014/002 and SAJ/2014/003' with an amount of '\$ 1,175.00'. The second row shows an incoming payment from 'INV/2017/0003' with an amount of '\$ 525.00'. The third row shows an incoming payment from 'INV/2017/0002' with an amount of '\$ 650.00'. The fourth row shows an incoming payment from 'INV/2017/0004' with an amount of '\$ 4,610.00'. The fifth row shows a bank fees entry with an amount of '\$ 32.58'. At the bottom of the table, there are buttons for 'Create Write-off' and 'Select a partner or choose a counterpart'.

Proof of Concept: 128 LOC



4

New Testing Framework

“ I don't like all these tests. That is way too much. It will slow you down.

”

someone above me...

New Testing Framework

Route /web/tests

Web Tests		
<input type="checkbox"/> Hide passed tests <input type="checkbox"/> Check for Globals <input type="checkbox"/> No try-catch		
Filter: <input type="text"/> Go		
QUnit 2.2.1; Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_6) Chrome/61.0.3163.91 Safari/537.36		
sort by time (desc)		
870 tests completed in 1377738 milliseconds, with 0 failed, 1 skipped, and 0 todo. 4225 assertions of 4225 passed, 0 failed.		
1. core > Registry: key set (1)	Rerun	10 ms
2. core > Registry: extension (2)	Rerun	0 ms
3. core > Registry: remain-linked (2)	Rerun	0 ms
4. core > Registry: multiget (1)	Rerun	0 ms
much testing!		
5. core > Registry: extended-multiget (1)	Rerun	1 ms
6. core > PY.eval: simple python expression (2)	Rerun	2 ms
7. core > PY.eval: not prefix (3)	Rerun	1 ms
8. core > PY.eval: strftime (3)	Rerun	3 ms
9. core > PY.eval: context_today (1)	Rerun	1 ms
10. core > PY.eval: timedelta.test_constructor (16)	Rerun	6 ms
11. core > PY.eval: timedelta.test_computations (28)	Rerun	11 ms
12. core > PY.eval: timedelta.test_basic_attributes (3)	Rerun	1 ms

New Testing Framework

- ▶ new views were developed with TDD
- ▶ (almost) pure QUnitjs
- ▶ powerful test helpers to mock RPCs
- ▶ can test async code synchronously!

New Testing Framework

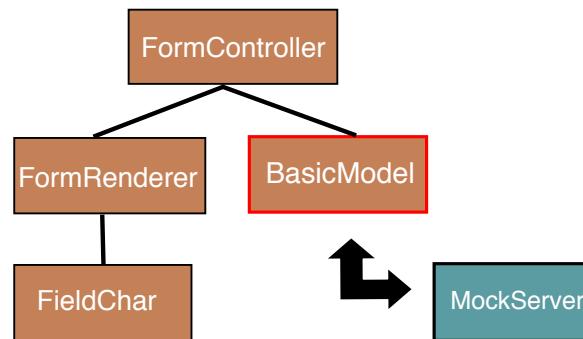
- ▶ new views were developed with TDD
- ▶ (almost) pure QUnitjs
- ▶ powerful test helpers to mock RPCs
- ▶ can test async code synchronously!

Benefits

- ▶ Executable specification
- ▶ much easier to accept PR
- ▶ much easier to refactor

Test Hero: MockServer

`mock_server.js`



Can mock: copy, create, default_get, fields_get, name_get, name_create, name_search, onchange, read, read_group, search_count, search_read, unlink, write, ...

```
QUnit.test('no html injection in char field', function (assert) {
    assert.expect(1);

    var form = createView({
        View: FormView,
        model: 'partner',
        data: this.data,
        arch: '<form string="Partners">' +
            '<field name="foo"/>' +
            '</form>',
        res_id: 1,
        viewOptions: { mode: 'edit' },
    });

    form.$('input')
        .val('<script>throw Error();</script>')
        .trigger('input');
    form.$buttons.find('.o_form_button_save').click();
    assert.strictEqual(
        form.$('.o_field_widget').text(),
        '<script>throw Error();</script>',
        'the value should have been properly escaped'
    );

    form.destroy();
});
```

A large, white, sans-serif number '5' is centered within a thick, light-gray circular outline. The entire graphic is set against a dark purple background.

5

Documentation and Guidelines

Documentation and Guidelines

Bad news

Did not have time to update main documentation...

Documentation and Guidelines

Bad news

Did not have time to update main documentation...

Good news

- ▶ finally committed to document core classes
- ▶ coding guidelines
- ▶ up-to-date doc online (extracted from docstring)

Uptodate html doc

Starting in v11 => doc extracted from docstrings

The screenshot shows a developer documentation page for the Odoo JS Framework. The top navigation bar includes links for User, Developer, API, Installation, White Papers, Legal, and a green 'START NOW' button. The main content area is titled 'Developer Doc / Javascript'. It displays the class definition for `AbstractController`, which extends `Widget`. The class has parameters: `parent` (Widget), `model` (`AbstractModel`), `renderer` (`AbstractRenderer`), and `params` (`AbstractControllerParams`). Below this, there are two methods: `start()` and `discardChanges([recordID])`. The `start()` method is described as simply rendering and updating the URL, with a return type of `Deferred`. The `discardChanges` method is described as discarding changes made on a record or by the controller, with a note that it returns a deferred. Parameters for `discardChanges` include `recordID` (string) and `Returns` (resolved if properly discarded, rejected otherwise). The return type for `discardChanges` is also `Deferred`.

Coding Guidelines

Code Issues 2,671 Pull requests 1,296 Wiki Insights ▾

Javascript coding guidelines

Quentin Smetz edited this page on Aug 21 · 12 revisions

From v11, we introduce a new coding standard for Odoo Javascript code. Here it is:

Use a Linter

- A sample configuration for ESLint:

```
{  
  "env": {  
    "commonjs": true,  
    "es6": true  
  }  
}
```

6

The Future

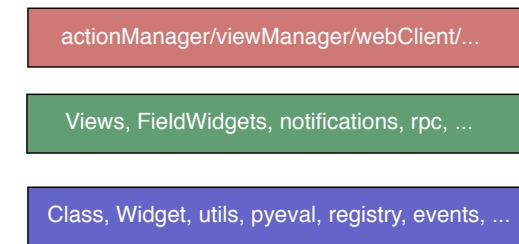
Current State

actionManager/viewManager/webClient/...

Views, FieldWidgets, notifications, rpc, ...

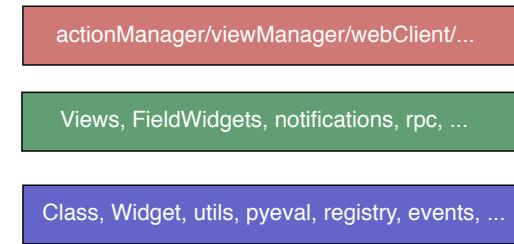
Class, Widget, utils, pyeval, registry, events, ...

Current State



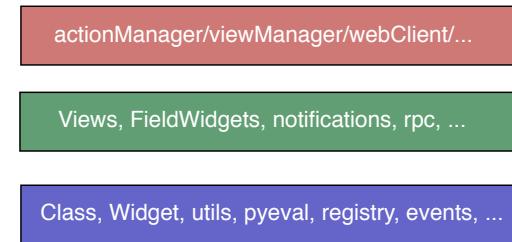
► bottom layer: mostly ok

Current State



- ▶ bottom layer: mostly ok
- ▶ medium layer: same

Current State



- ▶ bottom layer: mostly ok
- ▶ medium layer: same
- ▶ top layer: vastly too complex

High priority

- ▶ massive simplification of top layer
- ▶ high quality developer documentation (JS)

High priority

- ▶ massive simplification of top layer
- ▶ high quality developer documentation (JS)

Medium priority

- ▶ add didMount and willUnmount to Widget (remove do_show)

High priority

- ▶ massive simplification of top layer
- ▶ high quality developer documentation (JS)

Medium priority

- ▶ add didMount and willUnmount to Widget (remove do_show)
- ▶ simplify basicmodel (+ perf improvements)

High priority

- ▶ massive simplification of top layer
- ▶ high quality developer documentation (JS)

Medium priority

- ▶ add didMount and willUnmount to Widget (remove do_show)
- ▶ simplify basicmodel (+ perf improvements)
- ▶ solve the field widget model extension problem

High priority

- ▶ massive simplification of top layer
- ▶ high quality developer documentation (JS)

Medium priority

- ▶ add didMount and willUnmount to Widget (remove do_show)
- ▶ simplify basicmodel (+ perf improvements)
- ▶ solve the field widget model extension problem
- ▶ remove dataset and other old cruft

High priority

- ▶ massive simplification of top layer
- ▶ high quality developer documentation (JS)

Medium priority

- ▶ add didMount and willUnmount to Widget (remove do_show)
- ▶ simplify basicmodel (+ perf improvements)
- ▶ solve the field widget model extension problem
- ▶ remove dataset and other old cruft
- ▶ allow EcmaScript 2015

High priority

- ▶ massive simplification of top layer
- ▶ high quality developer documentation (JS)

Medium priority

- ▶ add didMount and willUnmount to Widget (remove do_show)
- ▶ simplify basicmodel (+ perf improvements)
- ▶ solve the field widget model extension problem
- ▶ remove dataset and other old cruft
- ▶ allow EcmaScript 2015
- ▶ improve error handling (logging/catching errors/recovery)

High priority

- ▶ massive simplification of top layer
- ▶ high quality developer documentation (JS)

Medium priority

- ▶ add didMount and willUnmount to Widget (remove do_show)
- ▶ simplify basicmodel (+ perf improvements)
- ▶ solve the field widget model extension problem
- ▶ remove dataset and other old cruft
- ▶ allow EcmaScript 2015
- ▶ improve error handling (logging/catching errors/recovery)
- ▶ ...

What makes a good framework?

What makes a good framework?

expressivity

What makes a good framework?

easy to understand

expressivity

What makes a good framework?

one obvious solutions for most problems

expressivity

easy to understand

API stability

What makes a good framework?

one obvious solutions for most problems

expressivity

easy to understand

API stability

What makes a good framework?

one obvious solutions for most problems

consistency

expressivity

easy to understand

API stability

What makes a good framework?

one obvious solutions for most problems

consistency

expressivity

easy to understand

Odoo v11 is much better!!!

Thank you for your attention...

Views: challenges/unsolved problems

- ▶ Field widget which needs to integrate with BasicModel
- ▶ Bad API when using field widgets standalone
- ▶ Optimizing RPCs/commands