An Introduction to Form Builder

A New Interface for Fields

NATHAN HAUG * DRUPALCON LONDON * AUGUST 25, 2011

Lullabot™

Consulting | Development | Training

Nathan Haug
"quicksketch" on d.o
Webform, Insert, Flag, FileField, ImageField

Lullabot™

Consulting | Development | Training

Nathan Haug

"quicksketch" on d.o

Webform, Insert, Flag, FileField, ImageField

original (view actual size)
600px
800px
thumbnail (view actual size)
75px
100px

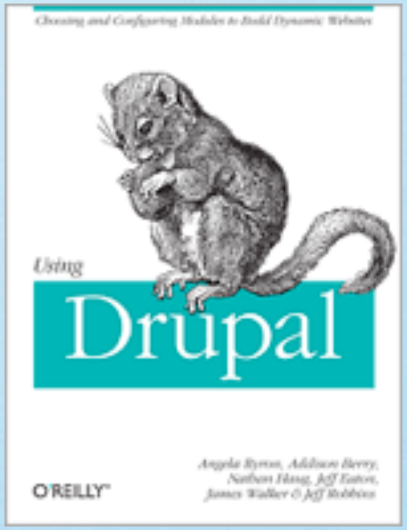
Lullabot™

Consulting | Development | Training

Nathan Haug

"quicksketch" on d.o

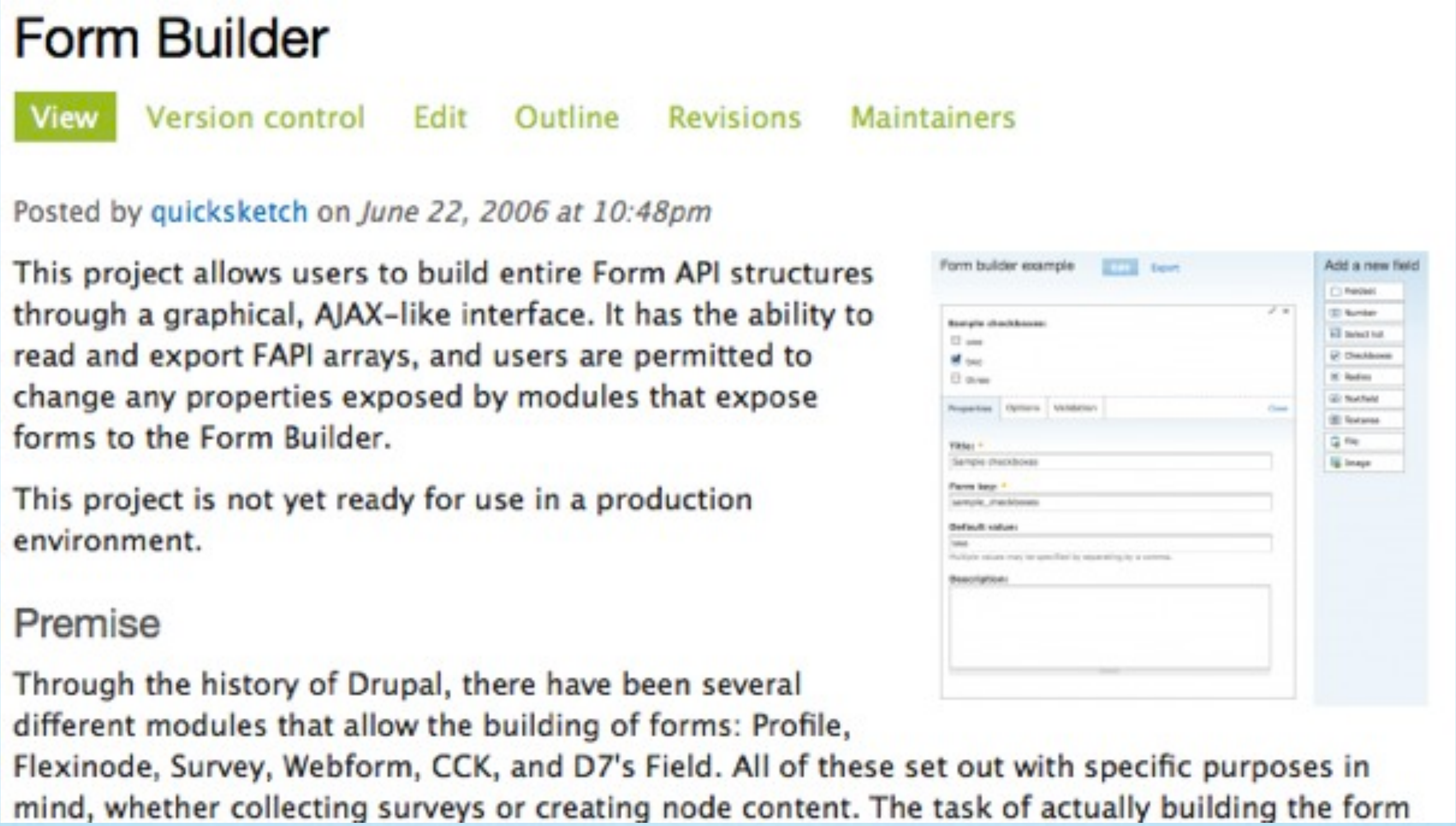
Webform, Insert, Flag, FileField, ImageField


Choosing and Configuring Modules to Build Dynamic Websites
Using
Drupal
O'REILLY
Angela Byron, Addison Berry, Nathan Haug, Jeff Eaton, James Walker & Jeff Robbins

drupalize.me

Get instant access to an unrivaled library of Drupal training from top-tier experts streaming to your computer, tablet, smart phone, & tv.

Form Builder History



Form Builder

Posted by quicksketch on *June 22, 2006 at 10:48pm*

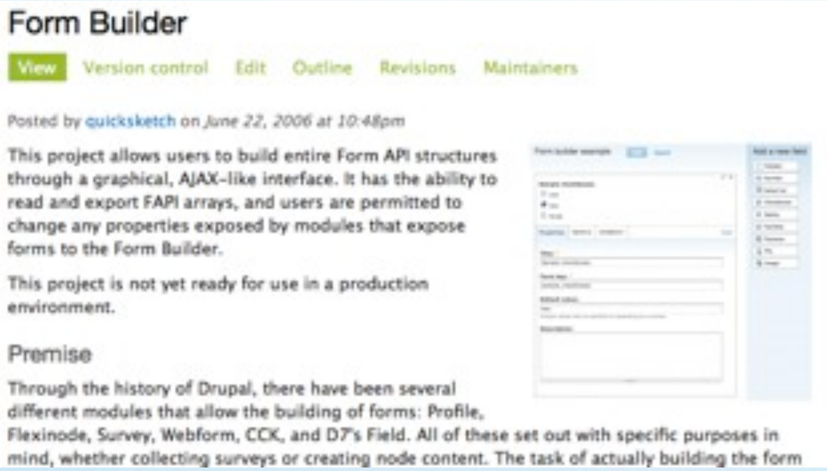
This project allows users to build entire Form API structures through a graphical, AJAX-like interface. It has the ability to read and export FAPI arrays, and users are permitted to change any properties exposed by modules that expose forms to the Form Builder.

This project is not yet ready for use in a production environment.

Premise

Through the history of Drupal, there have been several different modules that allow the building of forms: Profile, Flexinode, Survey, Webform, CCK, and D7's Field. All of these set out with specific purposes in mind, whether collecting surveys or creating node content. The task of actually building the form

Form Builder History



Form Builder

Posted by quicksketch on *June 22, 2006 at 10:48pm*

This project allows users to build entire Form API structures through a graphical, AJAX-like interface. It has the ability to read and export FAPI arrays, and users are permitted to change any properties exposed by modules that expose forms to the Form Builder.

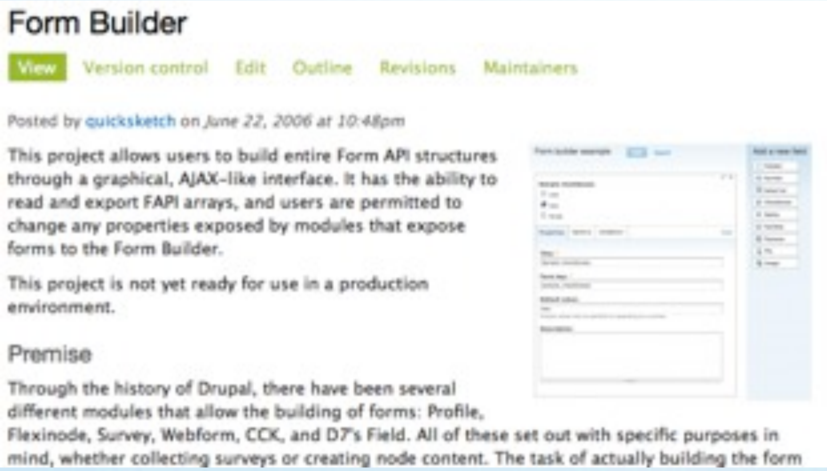
This project is not yet ready for use in a production environment.

Premise

Through the history of Drupal, there have been several different modules that allow the building of forms: Profile, Flexinode, Survey, Webform, CCK, and D7's Field. All of these set out with specific purposes in mind, whether collecting surveys or creating node content. The task of actually building the form

- http://drupal.org/project/form_builder

Form Builder History



Form Builder

Posted by quicksketch on *June 22, 2006 at 10:48pm*

This project allows users to build entire Form API structures through a graphical, AJAX-like interface. It has the ability to read and export FAPI arrays, and users are permitted to change any properties exposed by modules that expose forms to the Form Builder.

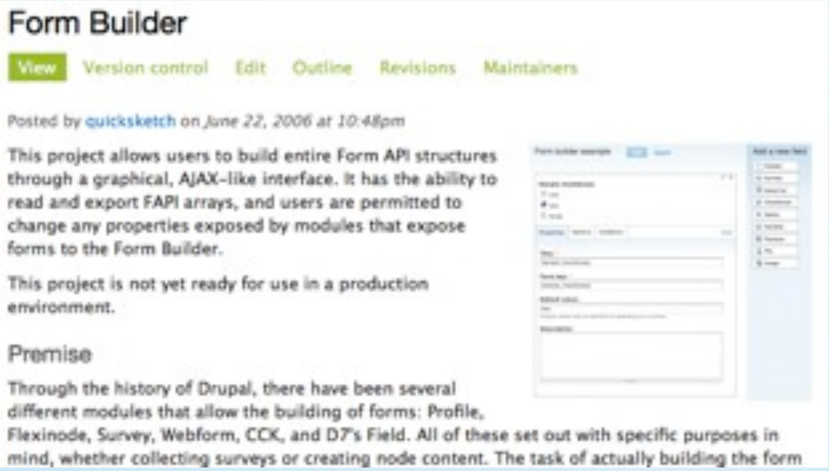
This project is not yet ready for use in a production environment.

Premise

Through the history of Drupal, there have been several different modules that allow the building of forms: Profile, Flexinode, Survey, Webform, CCK, and D7's Field. All of these set out with specific purposes in mind, whether collecting surveys or creating node content. The task of actually building the form

- http://drupal.org/project/form_builder
- Originally a Google Summer of Code Project

Form Builder History



Form Builder

Posted by quicksketch on *June 22, 2006 at 10:48pm*

This project allows users to build entire Form API structures through a graphical, AJAX-like interface. It has the ability to read and export FAPI arrays, and users are permitted to change any properties exposed by modules that expose forms to the Form Builder.

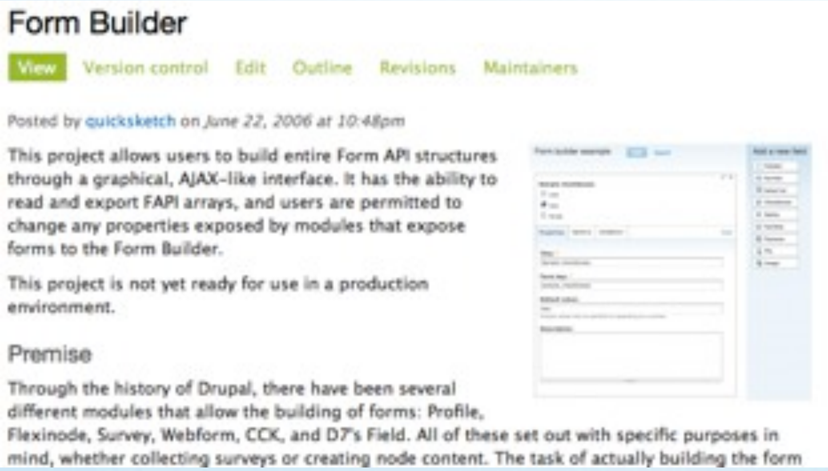
This project is not yet ready for use in a production environment.

Premise

Through the history of Drupal, there have been several different modules that allow the building of forms: Profile, Flexinode, Survey, Webform, CCK, and D7's Field. All of these set out with specific purposes in mind, whether collecting surveys or creating node content. The task of actually building the form

- http://drupal.org/project/form_builder
- Originally a Google Summer of Code Project
- 2008 - Rewritten in its current incarnation

Form Builder History



Form Builder

Posted by quicksketch on *June 22, 2006 at 10:48pm*

This project allows users to build entire Form API structures through a graphical, AJAX-like interface. It has the ability to read and export FAPI arrays, and users are permitted to change any properties exposed by modules that expose forms to the Form Builder.

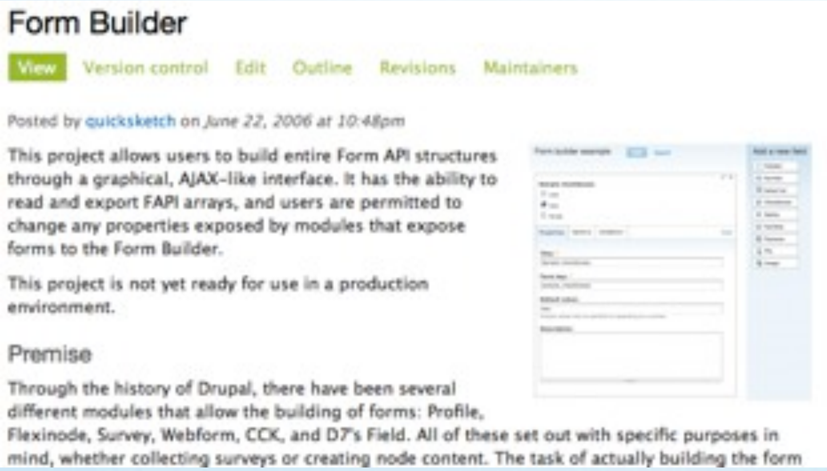
This project is not yet ready for use in a production environment.

Premise

Through the history of Drupal, there have been several different modules that allow the building of forms: Profile, Flexinode, Survey, Webform, CCK, and D7's Field. All of these set out with specific purposes in mind, whether collecting surveys or creating node content. The task of actually building the form

- http://drupal.org/project/form_builder
- Originally a Google Summer of Code Project
- 2008 - Rewritten in its current incarnation
- 2010 - Ported to D7 and included in Acquia Gardens

Form Builder History



Form Builder

Posted by quicksketch on *June 22, 2006 at 10:48pm*

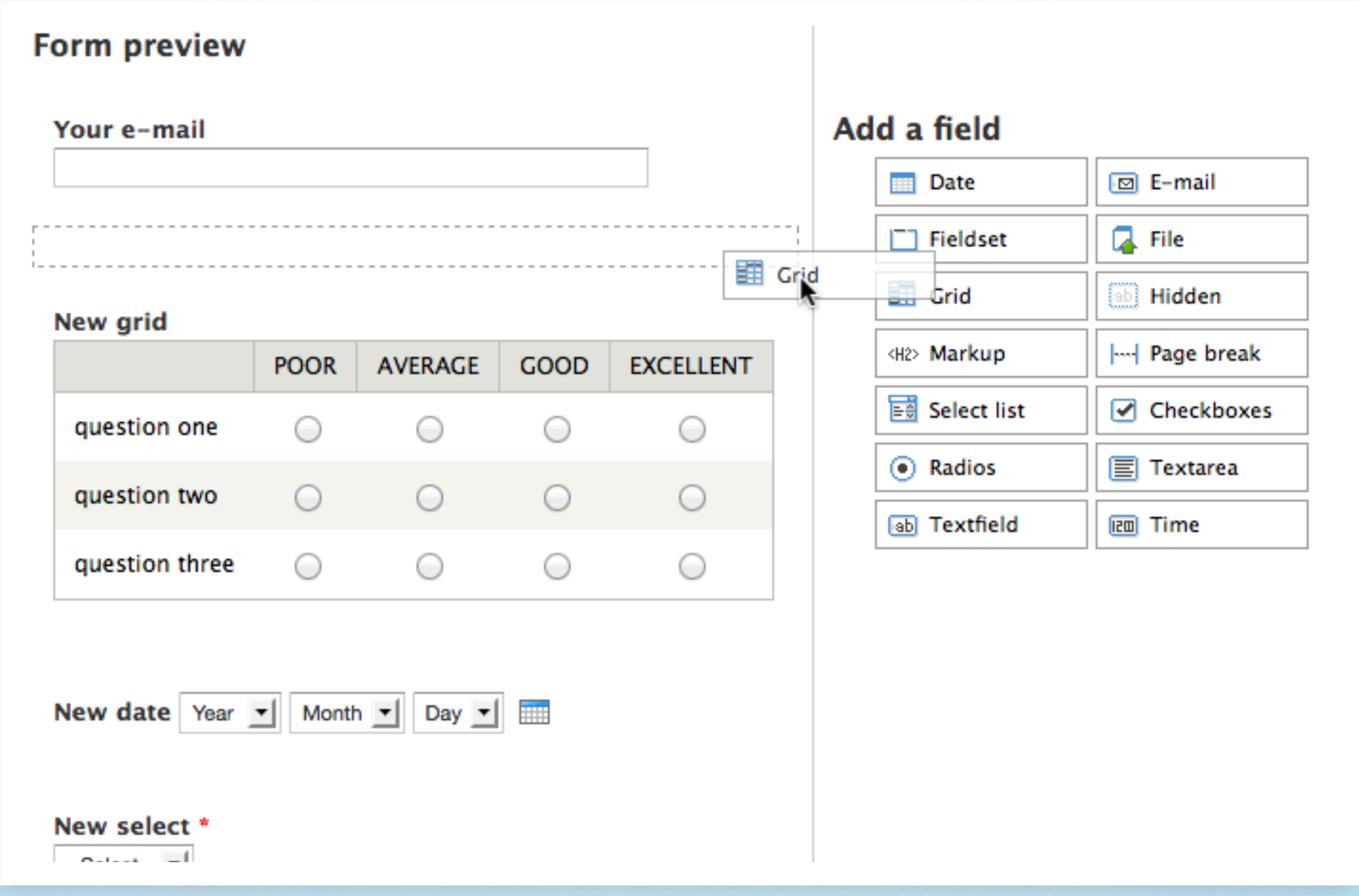
This project allows users to build entire Form API structures through a graphical, AJAX-like interface. It has the ability to read and export FAPI arrays, and users are permitted to change any properties exposed by modules that expose forms to the Form Builder.

This project is not yet ready for use in a production environment.

Premise

Through the history of Drupal, there have been several different modules that allow the building of forms: Profile, Flexinode, Survey, Webform, CCK, and D7's Field. All of these set out with specific purposes in mind, whether collecting surveys or creating node content. The task of actually building the form

- http://drupal.org/project/form_builder
- Originally a Google Summer of Code Project
- 2008 - Rewritten in its current incarnation
- 2010 - Ported to D7 and included in Acquia Gardens
- 2011 - Expanded and fully supported in Webform



Form preview

Your e-mail

New grid

	POOR	AVERAGE	GOOD	EXCELLENT
question one	○	○	○	○
question two	○	○	○	○
question three	○	○	○	○

New date Year Month Day

New select *

Demo of Webform Today

More on Webform (DrupalCon Chicago): http://www.archive.org/details/webform_3

The Way It Works

FormAPI array editor

The Way It Works

FormAPI array editor

```
$form['title'] = array(
  '#type' => 'textfield',
  '#title' => t('Title'),
  '#description' => t('Please enter the title for your post.'),
  '#maxlength' => 140,
  '#required' => TRUE,
  '#weight' => -10,
);
```

The Way It Works

~~FormAPI~~ Renderable array editor

The Way It Works

~~FormAPI~~ Renderable array editor

```
$node->content['title'] = array(
  '#type' => 'markup',
  '#title' => t('Title'),
  '#label_display' => 'hidden',
  '#markup' => 'Sample title',
  '#weight' => -10,
);
```

The Way It Works

Renderable array editor

The Way It Works

Renderable array editor

- Call form_builder_interface($form_builder_id, $type)

The Way It Works

Renderable array editor

- Call form_builder_interface($form_builder_id, $type)
  - Retrieve a renderable array through a module responding to hook_form_builder_load(). This function may just call drupal_get_form() or node_view()

The Way It Works

Renderable array editor

- Call form_builder_interface($form_builder_id, $type)
  - Retrieve a renderable array through a module responding to hook_form_builder_load(). This function may just call drupal_get_form() or node_view()
- Let Form Builder manipulate properties it knows about

The Way It Works

Renderable array editor

- Call form_builder_interface($form_builder_id, $type)
  - Retrieve a renderable array through a module responding to hook_form_builder_load(). This function may just call drupal_get_form() or node_view()
- Let Form Builder manipulate properties it knows about
- Use a separate form to actually save all the changes by loading the Form Builder renderable from cache

Form Builder Hooks

Form Builder Hooks

- hook_form_builder_types()

Form Builder Hooks

- hook_form_builder_types()
  - Returns an array of form types (i.e. "webform", "entity", etc.) that contain a list of element types that are editable (i.e. "textfield", "checkboxes", etc.) Also determines palette grouping and whether fields are "unique".

Form Builder Hooks

- hook_form_builder_types()
  - Returns an array of form types (i.e. "webform", "entity", etc.) that contain a list of element types that are editable (i.e. "textfield", "checkboxes", etc.) Also determines palette grouping and whether fields are "unique".
- hook_form_builder_properties()

Form Builder Hooks

- hook_form_builder_types()
  - Returns an array of form types (i.e. "webform", "entity", etc.) that contain a list of element types that are editable (i.e. "textfield", "checkboxes", etc.) Also determines palette grouping and whether fields are "unique".
- hook_form_builder_properties()
  - Define renderable properties (i.e #title) and forms to edit that property.

Form Builder Hooks

- hook_form_builder_types()
  - Returns an array of form types (i.e. "webform", "entity", etc.) that contain a list of element types that are editable (i.e. "textfield", "checkboxes", etc.) Also determines palette grouping and whether fields are "unique".
- hook_form_builder_properties()
  - Define renderable properties (i.e #title) and forms to edit that property.
- hook_form_builder_load($form_builder_type, $form_builder_id)

Form Builder Hooks

- hook_form_builder_types()
  - Returns an array of form types (i.e. "webform", "entity", etc.) that contain a list of element types that are editable (i.e. "textfield", "checkboxes", etc.) Also determines palette grouping and whether fields are "unique".
- hook_form_builder_properties()
  - Define renderable properties (i.e #title) and forms to edit that property.
- hook_form_builder_load($form_builder_type, $form_builder_id)
  - Load a complete FAPI array based on a form type and ID. Called when form_builder_interface() is invoked.

Form Builder Hooks

Form Builder Hooks

- hook_form_builder_add_element_alter(&$element, $form_type, $form_id)

Form Builder Hooks

- hook_form_builder_add_element_alter(&$element, $form_type, $form_id)
  - Change elements right after its been dragged out of the palette (e.g. to assign it an ID of some kind).

Form Builder Hooks

- hook_form_builder_add_element_alter(&$element, $form_type, $form_id)
  - Change elements right after its been dragged out of the palette (e.g. to assign it an ID of some kind).
- hook_form_builder_preview_alter(&$element, $form_type, $form_id)

Form Builder Hooks

- hook_form_builder_add_element_alter(&$element, $form_type, $form_id)
  - Change elements right after its been dragged out of the palette (e.g. to assign it an ID of some kind).
- hook_form_builder_preview_alter(&$element, $form_type, $form_id)
  - Essentially a pre-render hook to modify the display of an element while in the preview (e.g. Show a #type = 'hidden' element as a text field)

Utility functions

Utility functions

- form_builder_get_element(&$form, $element_id)

Utility functions

- form_builder_get_element(&$form, $element_id)
    - $element_id comes from $element['#form_builder']['element_id']

Utility functions

- form_builder_get_element(&$form, $element_id)
  - $element_id comes from $element['#form_builder']['element_id']
- form_builder_set_element(&$form, $element)

Utility functions

- form_builder_get_element(&$form, $element_id)
  - $element_id comes from $element['#form_builder']['element_id']
- form_builder_set_element(&$form, $element)
- form_builder_cache_difference($form_type, $form_id, $exclude_weight = TRUE)

Utility functions

- form_builder_get_element(&$form, $element_id)
  - $element_id comes from $element['#form_builder']['element_id']
- form_builder_set_element(&$form, $element)
- form_builder_cache_difference($form_type, $form_id, $exclude_weight = TRUE)
  - Get a list of changed elements only; used during save operations.

Saving your changes

Saving your changes

```
function form_builder_field_entity_page($entity_type, $bundle) {
  module_load_include('inc', 'form_builder', 'includes/form_builder.admin');

  $output = array();
  $output['form_builder_interface'] = form_builder_interface('mymodule', $form_id);
  $output['form_builder_save_form'] = drupal_get_form('mymodule_save_form');
  return $output;
}
```

Saving your changes

```
function form_builder_field_entity_page($entity_type, $bundle) {
  module_load_include('inc', 'form_builder', 'includes/form_builder.admin');

  $output = array();
  $output['form_builder_interface'] = form_builder_interface('mymodule', $form_id);
  $output['form_builder_save_form'] = drupal_get_form('mymodule_save_form');
  return $output;
}
```

- mymodule_form_builder_load($form_builder_id, $type = NULL)

Saving your changes

```
function form_builder_field_entity_page($entity_type, $bundle) {
  module_load_include('inc', 'form_builder', 'includes/form_builder.admin');

  $output = array();
  $output['form_builder_interface'] = form_builder_interface('mymodule', $form_id);
  $output['form_builder_save_form'] = drupal_get_form('mymodule_save_form');
  return $output;
}
```

- mymodule_form_builder_load($form_builder_id, $type = NULL)

```
function mymodule_save_form($form, $form_state) {

  $form['submit'] = array(
    '#type' => 'submit',
    '#value' => t('Save'),
    '#weight' => -1,
  );

  return $form;
}
```

Saving your changes

```
function form_builder_field_entity_page($entity_type, $bundle) {
  module_load_include('inc', 'form_builder', 'includes/form_builder.admin');

  $output = array();
  $output['form_builder_interface'] = form_builder_interface('mymodule', $form_id);
  $output['form_builder_save_form'] = drupal_get_form('mymodule_save_form');
  return $output;
}
```

- mymodule_form_builder_load($form_builder_id, $type = NULL)

```
function mymodule_save_form($form, $form_state) {

  $form['submit'] = array(
    '#type' => 'submit',
    '#value' => t('Save'),
    '#weight' => -1,
  );

  return $form;
}
```

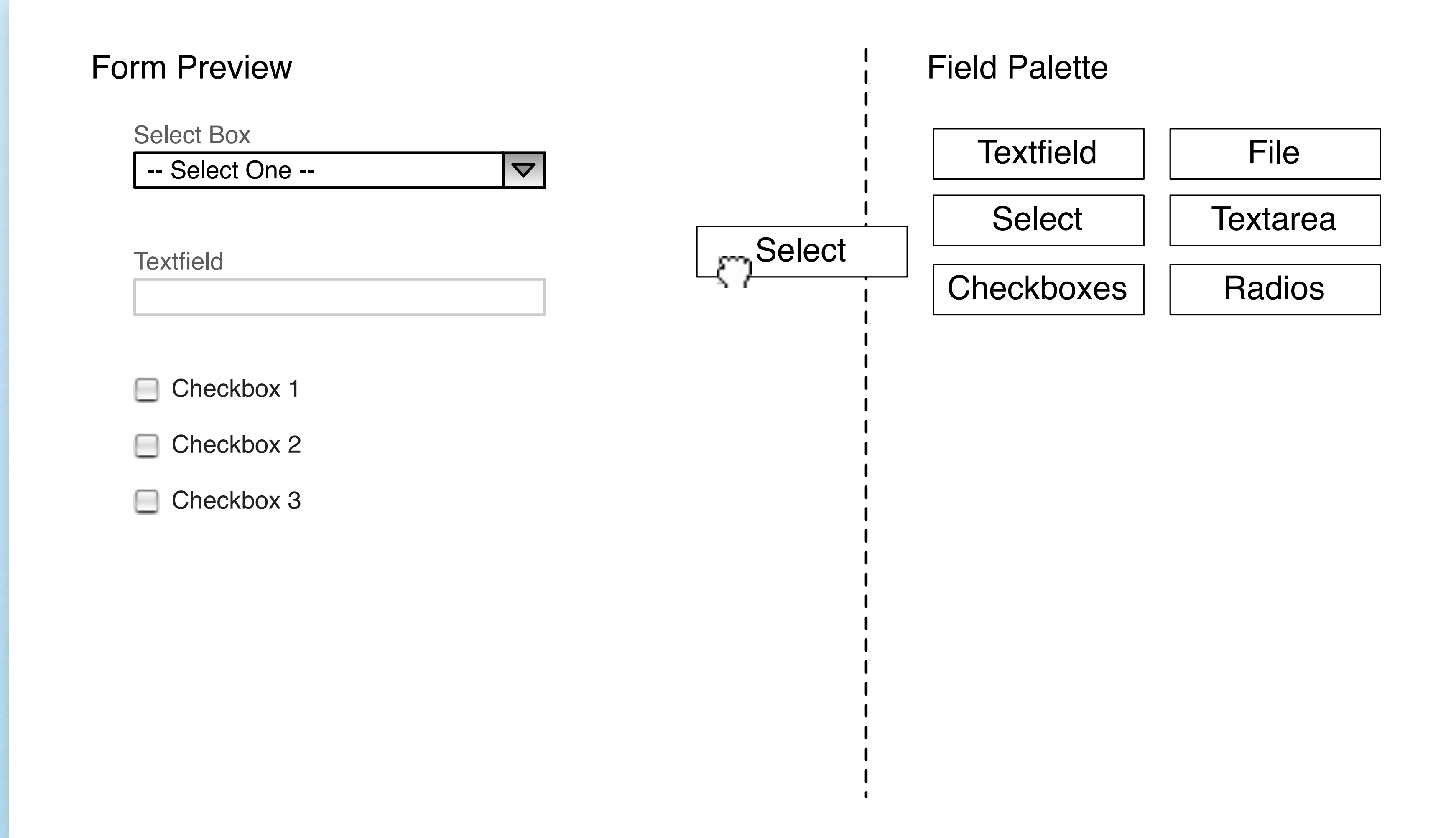
Saving your changes

```
function mymodule_save_form_submit($form, &$form_state) {
  // The entire renderable array with all the changes.
  $changed_form = form_builder_cache_load('myform', $type);

  // An array of just elements that have been modified.
  $differences = form_builder_cache_difference('myform', $type);

  foreach ($differences as $element_id => $element) {
    $original = $element['original'];
    $modified = $element['modified'];
    // Save the changes for the $modified element here.
    // db_insert/update, etc.
  }
}
```

Architectural Questions?

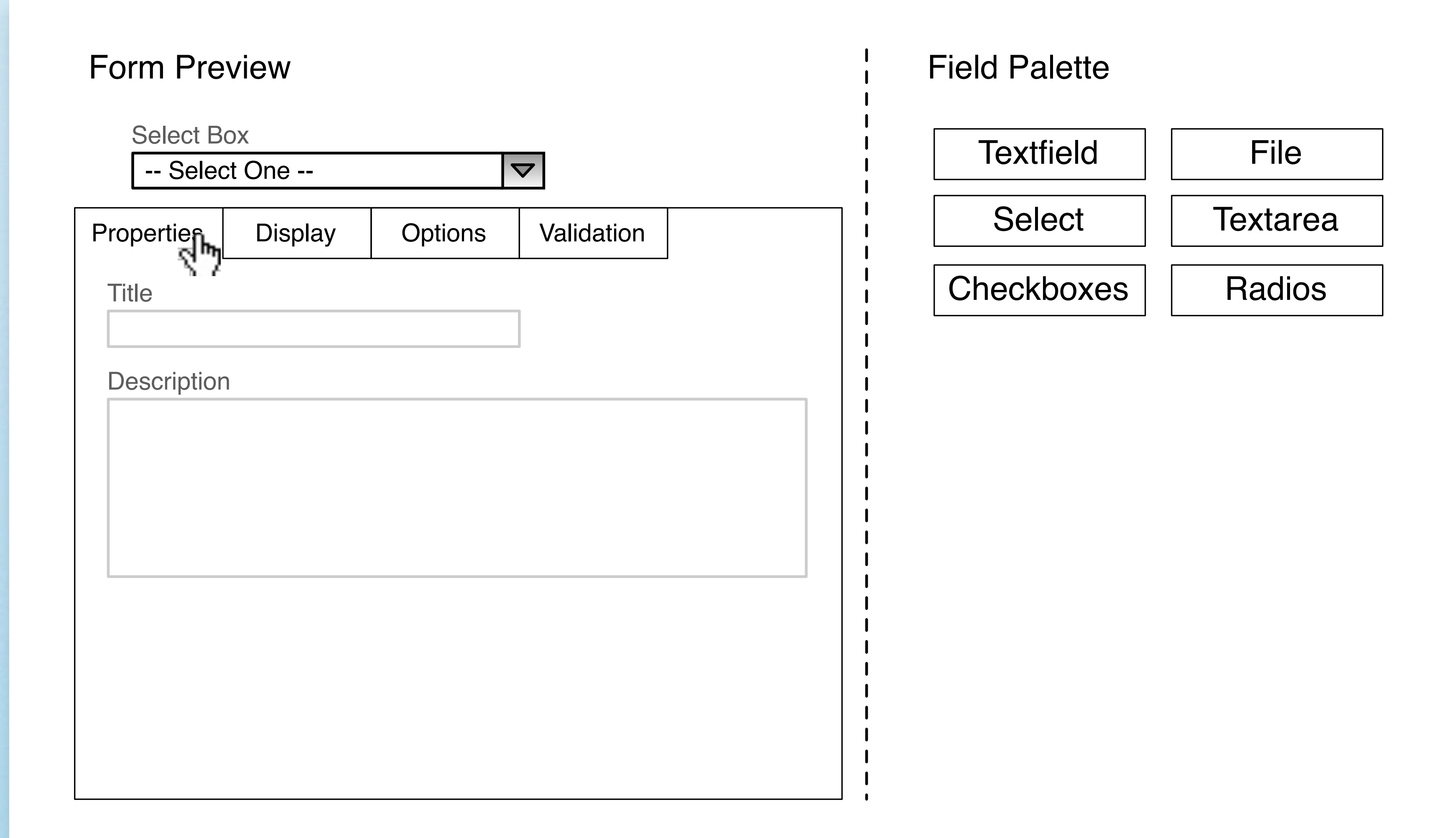
The Potential for "Fields"

Option 1: Stick with "Widget"-first palette



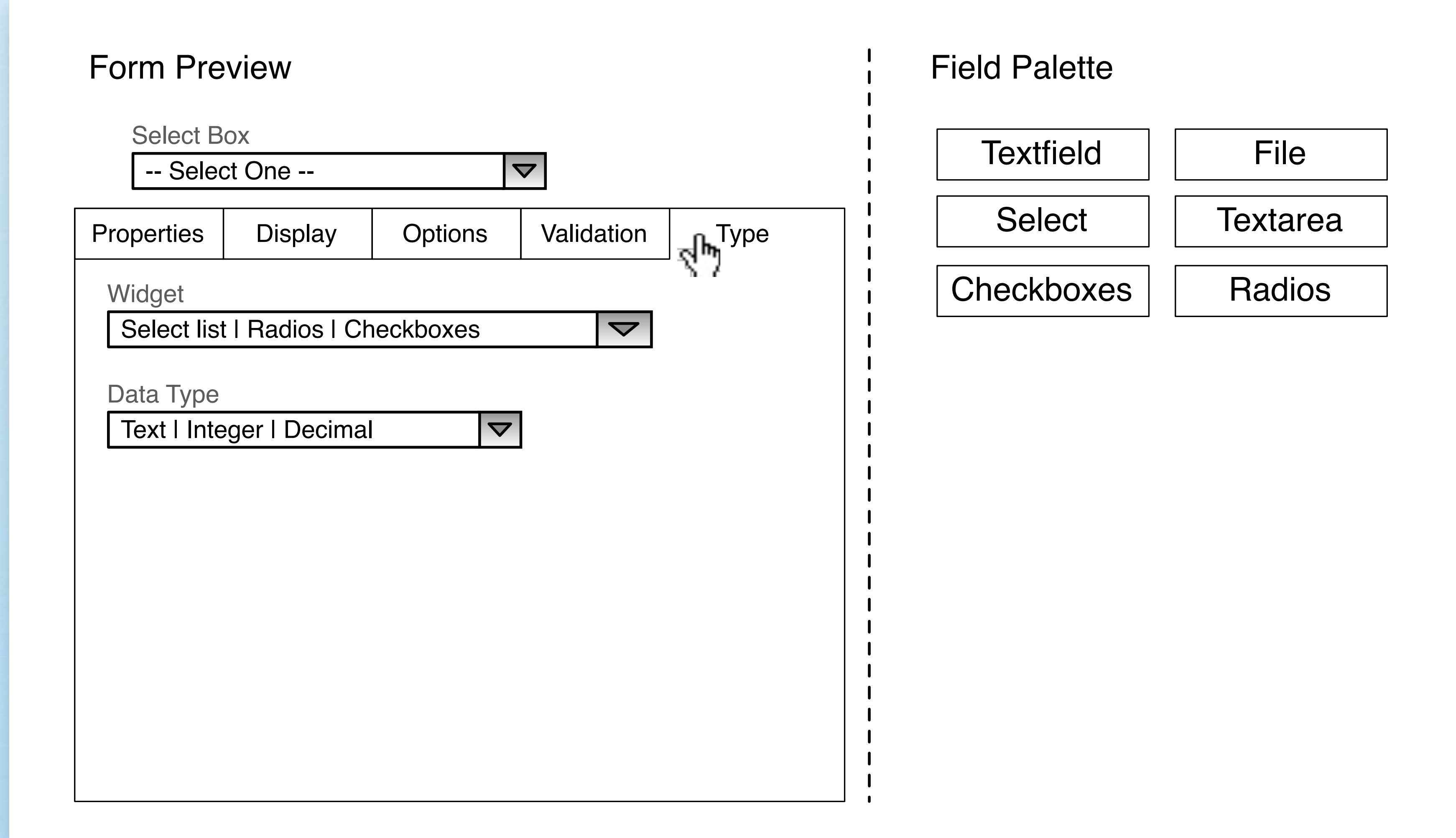
The Potential for "Fields"

Option 1: Which edits a field just like Webform



The Potential for "Fields"

Option 1: But adds a new tab for Widget and Data type



Form Preview

Select Box: -- Select One --

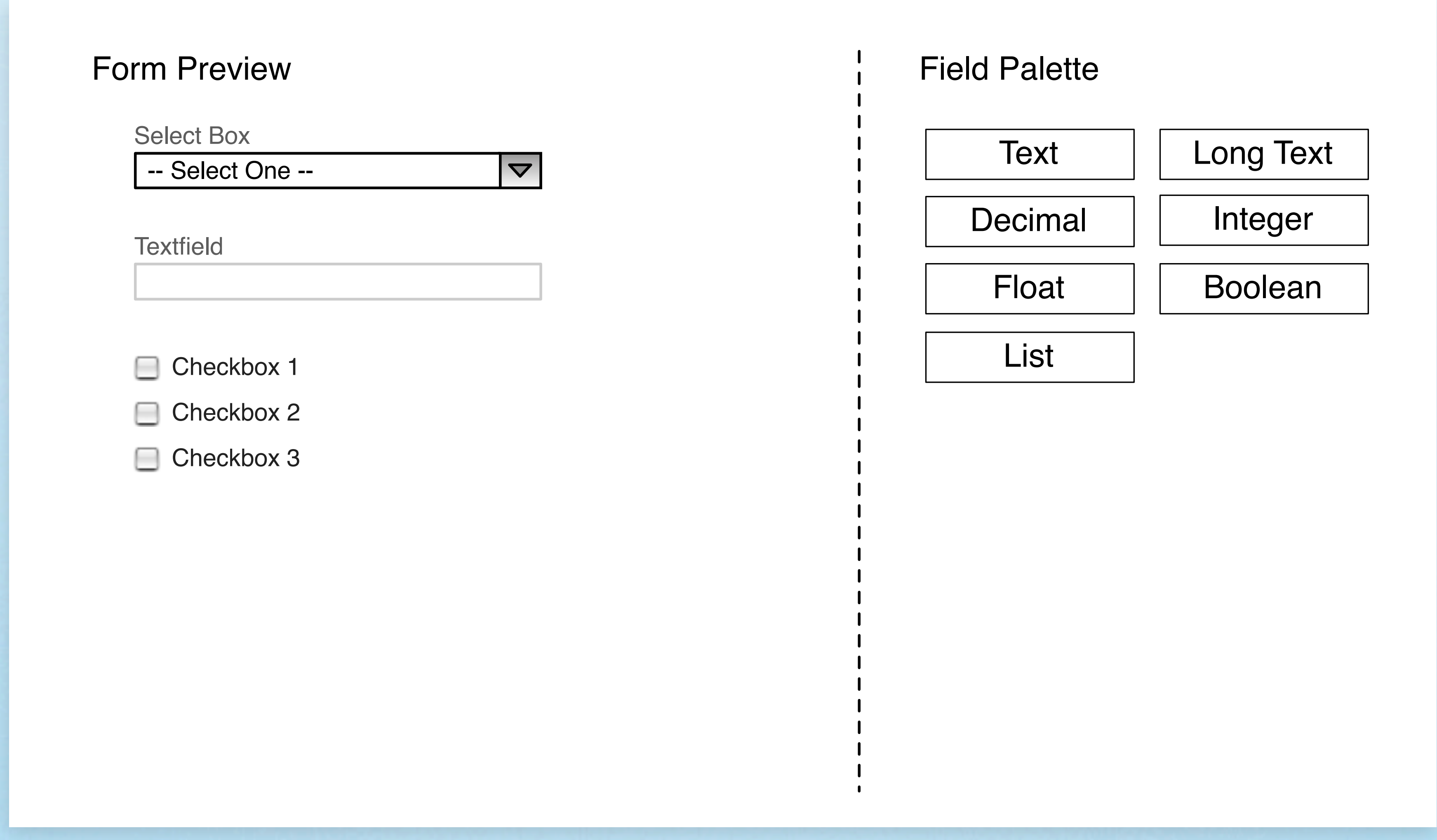
Widget: Select list | Radios | Checkboxes

Data Type: Text | Integer | Decimal

Field Palette

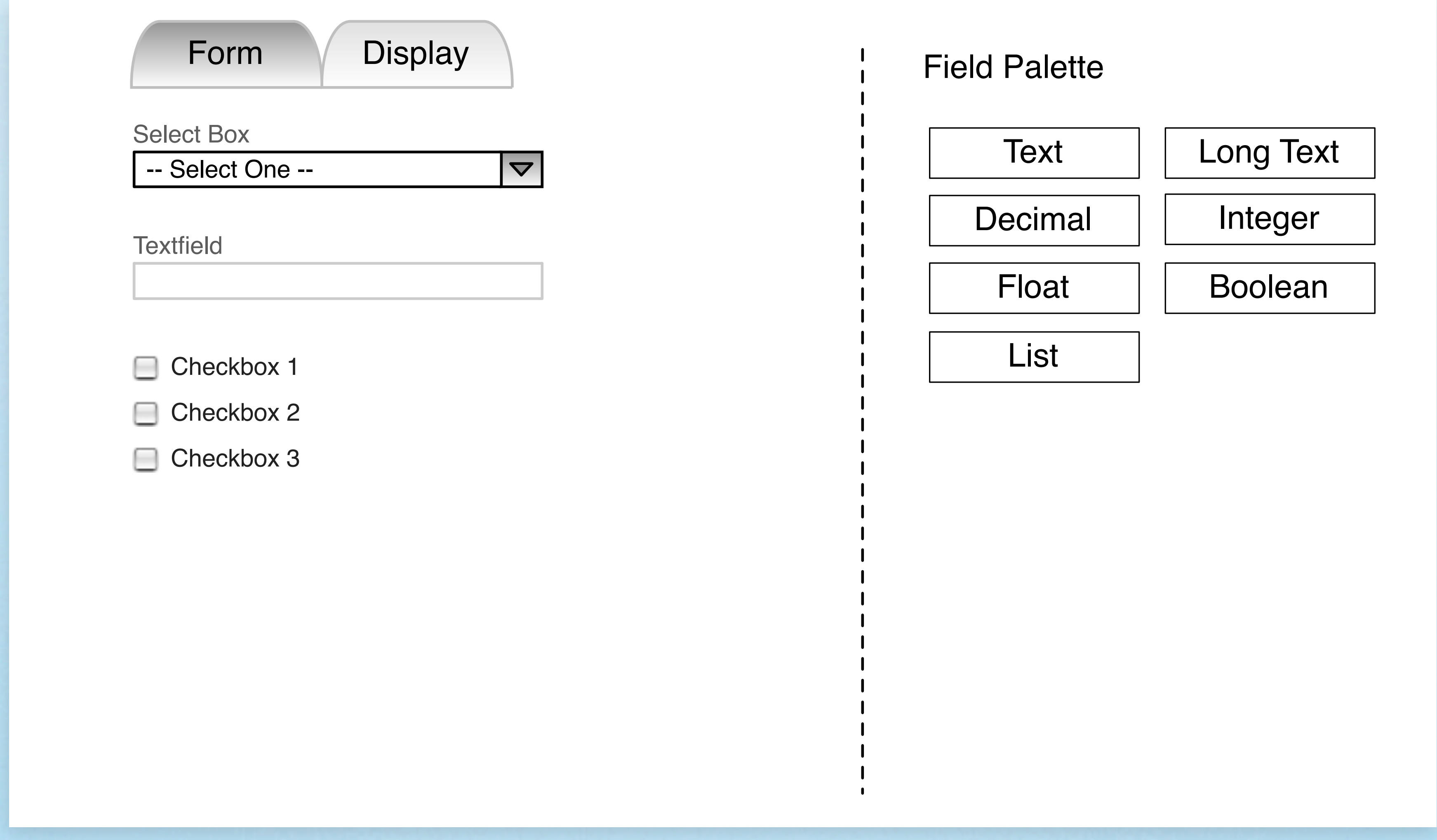
The Potential for "Fields"

Option 2: Replace widget palette with Data type



The Potential for "Fields"

Option 2: Has dual-purpose for forms **and** display



Select Box

-- Select One --

Textfield

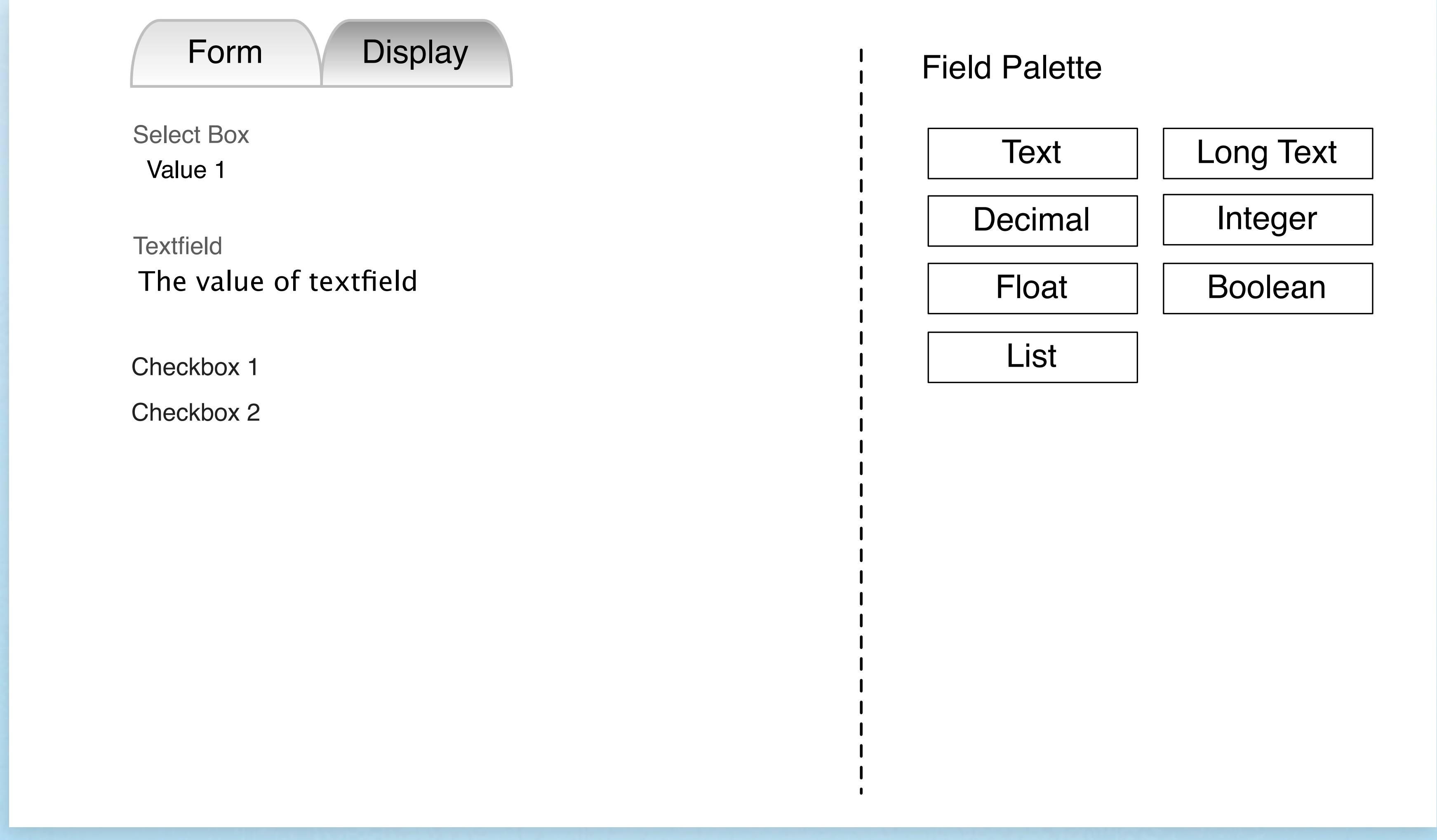
□ Checkbox 1

□ Checkbox 2

□ Checkbox 3

The Potential for "Fields"

Option 2: Has dual-purpose for forms **and** display



Select Box
Value 1

Textfield
The value of textfield

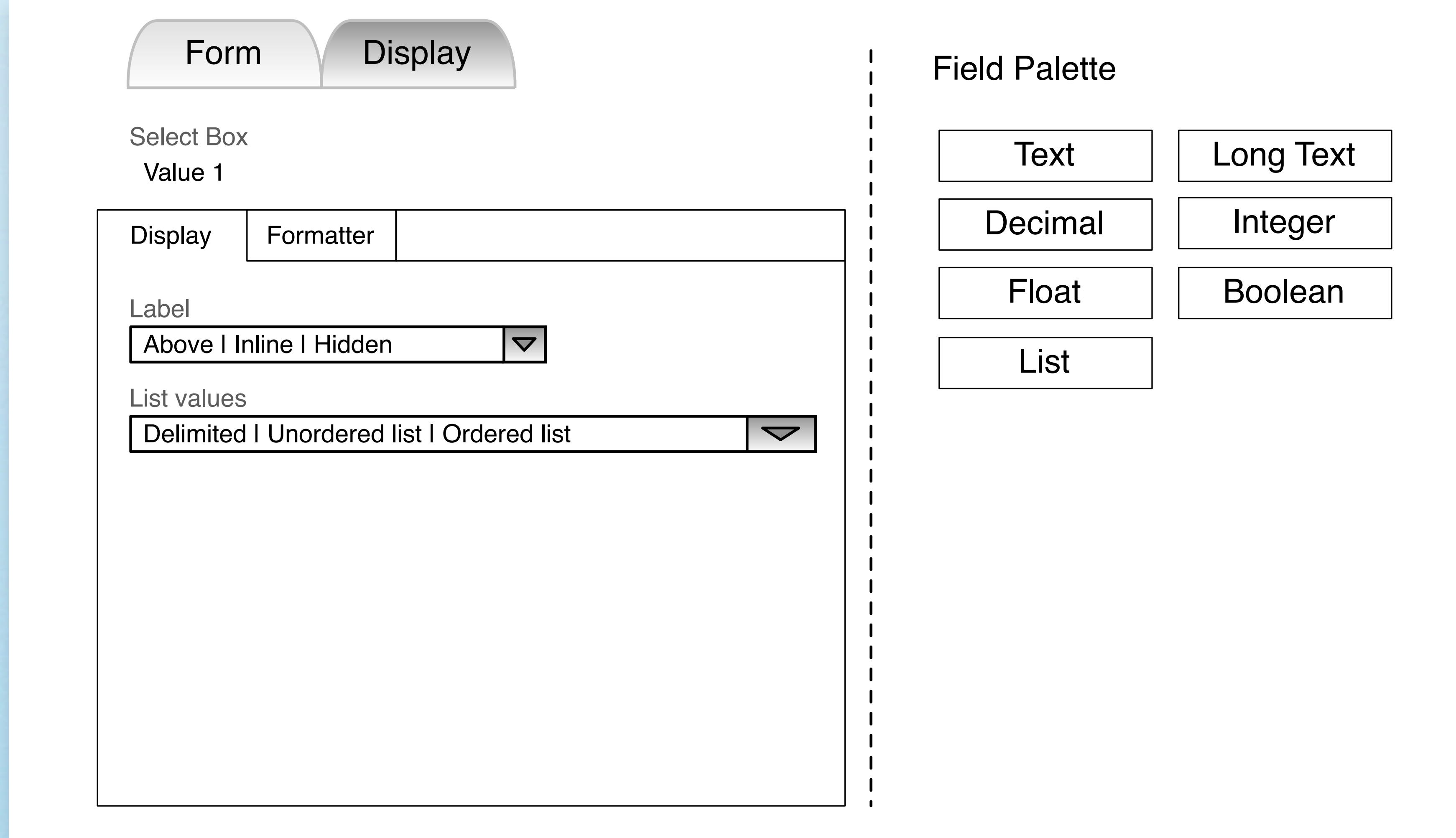
Checkbox 1

Checkbox 2

Field Palette

The Potential for "Fields"

Option 2: Configure formatter options for front-end



Select Box
Value 1

Label
Above | Inline | Hidden

List values
Delimited | Unordered list | Ordered list

Pre–requisites for Drupal core

Pre–requisites for Drupal core

- Each field must have a discrete function for generating preview (authoring information, title, publishing options, revision information).

Pre-requisites for Drupal core

- Each field must have a discrete function for generating preview (authoring information, title, publishing options, revision information).
- Each field must be able to provide a set of defaults, including label, default, list of options (in the case of radio buttons), etc.

Pre-requisites for Drupal core

- Each field must have a discrete function for generating preview (authoring information, title, publishing options, revision information).
- Each field must be able to provide a set of defaults, including label, default, list of options (in the case of radio buttons), etc.
- Static data object storage from CTools

Final Questions?

Locate this session on the DrupalCon London website:
http://london2011.drupal.org/conference/schedule/thursday

Click the "Evaluate this session" link.

Thanks!