# Super Fast Application Development with Mura CMS

# What’s Up?

- Build a fully functional application using Mura ORM & Mura.js

# Target Audience

- ColdFusion/CFML Developer
- Using Mura CMS
- Familiar with
  - .CFCs
  - Mura’s base objects (content, user, etc.)
  - Popular SQL DBMS
- May (or may not) have used ColdFusion’s ORM

# What is Mura CMS?

- Open source Content Management System (CMS)
- Let users add/edit content, while you build applications
- Used by clients large & small

# Mura Display Objects

- Self-contained display
  - Portable
  - Distributable (via plugins)
- Could be an entire application!

# The Mura Scope

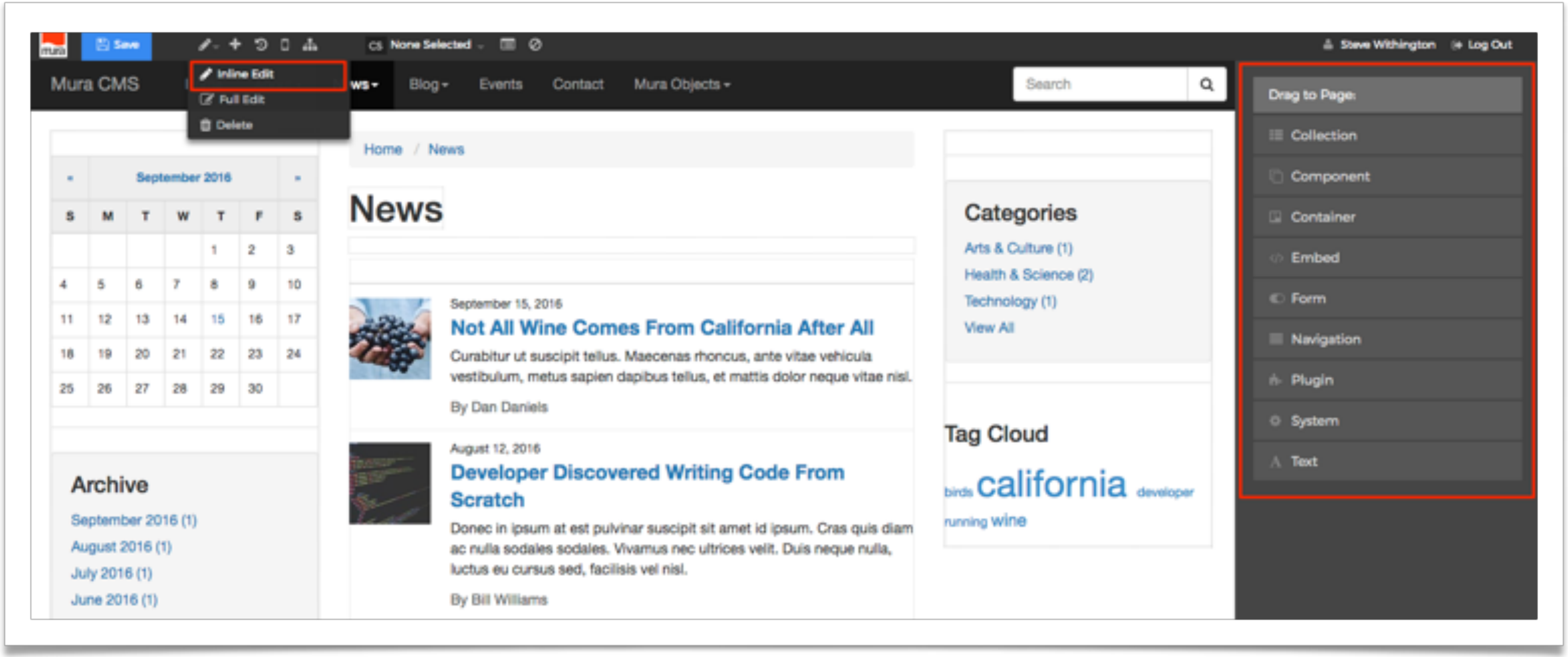
- A wrapper for interacting with Mura CMS
  - Access data
  - Create/modify data
  - Helper methods
- “`Mura`” or “`m`” and still supports “`$`”

# Mura Display Objects

Part 1

# Mura Display Objects

- Managed via front-end only (*in Mura v7+*)



Home / News

| « | | | September 2016 | | | » |
|---|---|---|---|---|---|---|
| S | M | T | W | T | F | S |
| | | | | 1 | 2 | 3 |
| 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 11 | 12 | 13 | 14 | 15 | 16 | 17 |
| 18 | 19 | 20 | 21 | 22 | 23 | 24 |
| 25 | 26 | 27 | 28 | 29 | 30 | |

Archive

September 2016 (1)
August 2016 (1)
July 2016 (1)
June 2016 (1)

News

September 15, 2016

Not All Wine Comes From California After All

Curabitur ut suscipit tellus. Maecenas rhoncus, ante vitae vehicula vestibulum, metus sapien dapibus tellus, et mattis dolor neque vitae nisl.

By Dan Daniels

August 12, 2016

Developer Discovered Writing Code From Scratch

Donec in ipsum at est pulvinar suscipit sit amet id ipsum. Cras quis diam ac nulla sodales sodales. Vivamus nec ultrices velit. Duis neque nulla, luctus eu cursus sed, facilisis vel nisl.

By Bill Williams

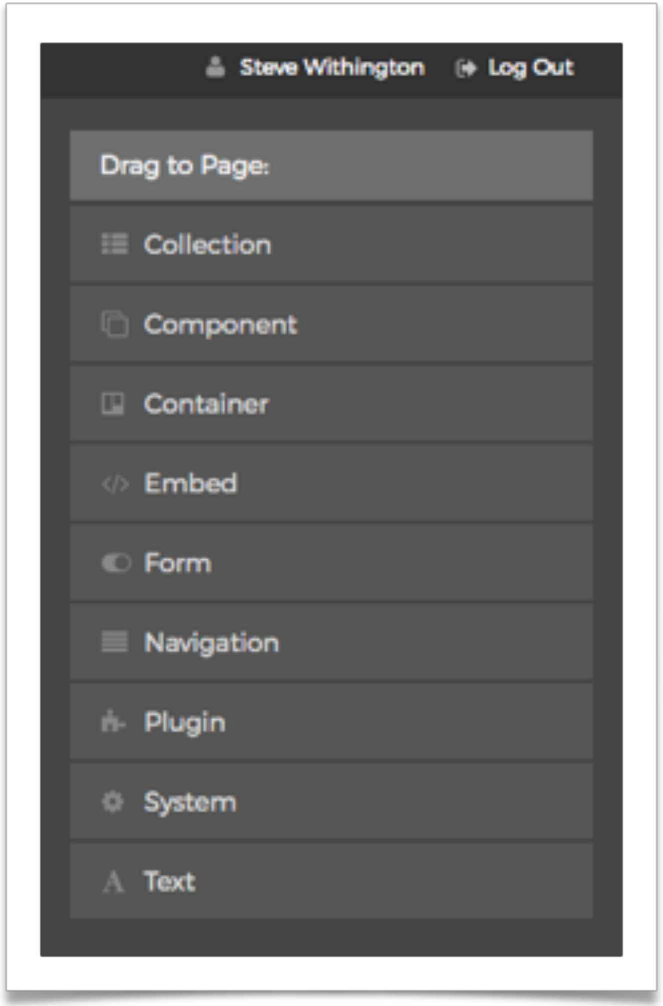
Categories

Arts & Culture (1)
Health & Science (2)
Technology (1)
View All

Tag Cloud

birds california developer running wine

# Mura Display Objects



https://github.com/stevewithington/
**murahelloworld**/tree/**cfml**

# What is **ORM**?

# What is ORM?

- **Popular SQL DBMS**
  - Only store/manipulate scalar values such as integers and strings organized within tables

# What is ORM?

- **Popular SQL DBMS**
  - Only store/manipulate scalar values such as integers and strings organized within tables
  - Object values must either be converted into groups of simpler values, or only use simple scalar values

# What is ORM?

- **Object-Relationtional Mapping** (ORM) **addresses the main issue with SQL DBMS**
  - Translates the logical representation of objects into an atomized form capable of being stored in a database, while preserving the properties of objects and their relationships so they can be reloaded as objects when needed

# What is ORM?

- Object-Relational Mapping (**ORM**) is essentially a “**virtual object database**”

# What is **Mura ORM**?

# What is Mura ORM?

- Mura ORM is a “**virtual object database**” that can be used within Mura

# What is Mura ORM?

- Mura ORM is a “**virtual object database**” that can be used within Mura
- Takes advantage of Di1 (dependency injection)
  - https://github.com/framework-one/di1

# What is Mura ORM?

- Mura ORM is a “**virtual object database**” that can be used within Mura
- Takes advantage of Di1 (dependency injection)
  - https://github.com/framework-one/di1
- Access & modify objects quickly & easily

# What is Mura ORM?

- Mura ORM is a “**virtual object database**” that can be used within Mura
- Takes advantage of Di1 (dependency injection)
  - https://github.com/framework-one/di1
- Access & modify objects quickly & easily
- No need for custom DAOs, class extensions, or database-vendor-specific CRUD statements

# What is Mura ORM?

- Mura ORM is a "**virtual object database**" that can be used within Mura
- Takes advantage of Di1 (dependency injection)
  - https://github.com/framework-one/di1
- Access & modify objects quickly & easily
- No need for custom DAOs, class extensions, or database-vendor-specific CRUD statements
- Similar to ColdFusion-based Hibernate ORM (*but not the same!*)

# Address Book

# Address Book

- Person

# Address Book

- Person
  - Zero or more phone numbers

# Address Book

- **Person**
  - Zero or more phone numbers
  - Zero or more addresses

# Address Book

- **Person** object with attributes/fields

# Address Book

- **Person** object with attributes/fields
  - Person’s name

# Address Book

- **Person** object with attributes/fields
  - Person’s name
  - List of phone numbers
    - **PhoneNumber** objects

# Address Book

- **Person** object with attributes/fields
  - Person’s name
  - List of phone numbers
    - **PhoneNumber** objects
  - List of addresses
    - **Address** objects

# Address Book

- The address book entry (Person) is treated as a single object

# Address Book

- The address book entry (Person) is treated as a single object
  - It can be referenced by a single variable containing a pointer to the object

# Address Book

- The address book entry (Person) is treated as a single object
  - It can be referenced by a single variable containing a pointer to the object
- Various helper methods can be associated with the object, such as a method to return the preferred phone number, the home address, and so on

https://github.com/stevewithington/
**muracontacts**/tree/**cfml**

# Mura.js

# Mura.js

- A lightweight utility to decouple dependency on jQuery

# Mura.js

- A lightweight utility to decouple dependency on jQuery
- A JS framework for interacting with the Mura JSON API

# Mura.js

- Familiar jQuery syntax

```
Mura(function(m) {
 m(‘.target’).each(function() {
  m(this).html(‘Mura found you!’);
 });
});
```

# Mura.js

- Baked-in **AJAX** support

```
Mura
  .ajax({
    type: ‘post’,
    url: ’https://domain.com/path/',
    data: { key: value },
    success: function(resp) { console.log(resp) },
    error: function(resp) { console.log(resp) }
  });
```

# Mura.js

- Baked-in AJAX support & **JS promises**

```
Mura
  .get(‘https://domain.com/path/')
  .then(function(resp) {
    // success
    Mura(‘#target’).html(resp.data.html);
  }, function(e) {
    // fail
  });
```

# Mura.js

- Baked-in AJAX support & JS promises

```
Mura
  .post(
    ‘https://domain.com/path/',
    { key: value }
  )
  .then(function(resp) {
    // success
  }, function(e) {
    // error
  });
```

# Mura.js

- JS promises can be stacked like Russian dolls

```
Mura.get(‘https://domain.com/path/')
  .then(function(resp) {
    // success
    Mura.get(‘https://domain.com?id=' + resp.id)
      .then(function(newresp) {
        Mura(‘#target’).html(newresp.html);
      });
  }, function(e) {
    // error
  });
```

# Mura.js

- Register event handlers

```
Mura(‘#mybutton’)
   .on(‘click’, function(e) {
     e.preventDefault();
     console.log(e);
   });
```

# Mura.js

- **Mura.DOMSelection** class
  - Wraps selected target via the Mura() method

# Mura.js

- **Mura.DOMSelection** class
  - Wraps selected target via the Mura() method
  - Allows you to handle selection as a *single object* or a *collection*

# Mura.js

- **Mura.DOMSelection** class
  - Wraps selected target via the Mura() method
  - Allows you to handle selection as a *single object* or a *collection*
    - **Single object**

      ```
      Mura('.target').html('Hello world!');
      ```

# Mura.js

- **Mura.DOMSelection** class
  - Wraps selected target via the Mura() method
  - Allows you to handle selection as a *single object* or a *collection*
    - **Single object**

      ```
      Mura('.target').html('Hello world!');
      ```

    - **Collection**

      ```
      Mura('.target').each(function() {
        Mura(this).html('Hello world!');
      });
      ```

# Mura.js

- Supported methods can be found under:

  `{SiteID}/js/src/`**`mura.domselection.js`**

# Mura.js

- **Load** entities

```
Mura
  .getEntity(‘person’)
  .loadBy(‘personid’, pid)
  .then(function(person) {
    // success
    console.log(person);
  }, function(person) {
    // fail
    console.log(person.get(‘errors’));
  });
```

# Mura.js

- **Create/Update** entities

```
Mura
  .getEntity(‘person’)
  .loadBy(‘personid’, pid)
  .then(function(person) {
    // load success
    person.set(‘namelast’, ‘Withington’)
      .save()
      .then(function(person) {
        // save success
      }, function(person) {
        // save fail
      });
  });
```

# Mura.js

- Delete entities

```
Mura
  .getEntity(‘person’)
  .loadBy(‘personid’, pid)
  .then(function(person) {
    // load success
    person
      .delete()
      .then(function(person) {
        // delete success
      }, function(person) {
        // delete fail
      });
  });
```

# Mura.js

- Feed API

```
Mura
  .getFeed(‘person’)
  .where()  // optional
  .prop(‘namelast’).isEQ(‘Levine’)
  .orProp(‘namelast’).beginsWith(‘Withing’)
  .getQuery()
  .then(function(people) {
    // success
    people.each(function(prsn, idx) {
      console.log(prsn.get(‘namefirst’) + ‘ ‘ + prsn.get(‘namelast’));
    });
  });
```

# Mura.js (Feed Methods)

```
Mura
  .getFeed([entityname])
  .where()
  .prop([property])
  .andProp([property])
  .orProp([property])
  .isEQ([criteria])
  .isNEQ([criteria])
  .isLT([criteria])
  .isLTE([criteria])
  .isGT([criteria])
  .isGTE([criteria])
  .containsValue([criteria])
  .null()
```

```
.beginsWith([criteria])
.endsWith([criteria])
.openGrouping()
.andOpenGrouping()
.closeGrouping()
.sort([property],[asc||desc])
.itemsPerPage([itemsPerPage])
.maxItems([maxItems])
.isIn([list criteria])
.isNotIn([list criteria])
.innerJoin([relatedEntity])
.leftJoin([relatedEntity])
.getQuery()
```

https://github.com/stevewithington/

**murahelloworld/**tree**/js**

https://github.com/stevewithington/
**muracontacts**/tree/**js**

# Q+A

# Resources

- http://www.getmura.com
- https://github.com/stevewithington/cfsummit-2016
- https://github.com/stevewithington/muracontacts
- https://github.com/stevewithington/murahelloworld
- https://groups.google.com/forum/#!forum/mura-cms-developers

Thank you!