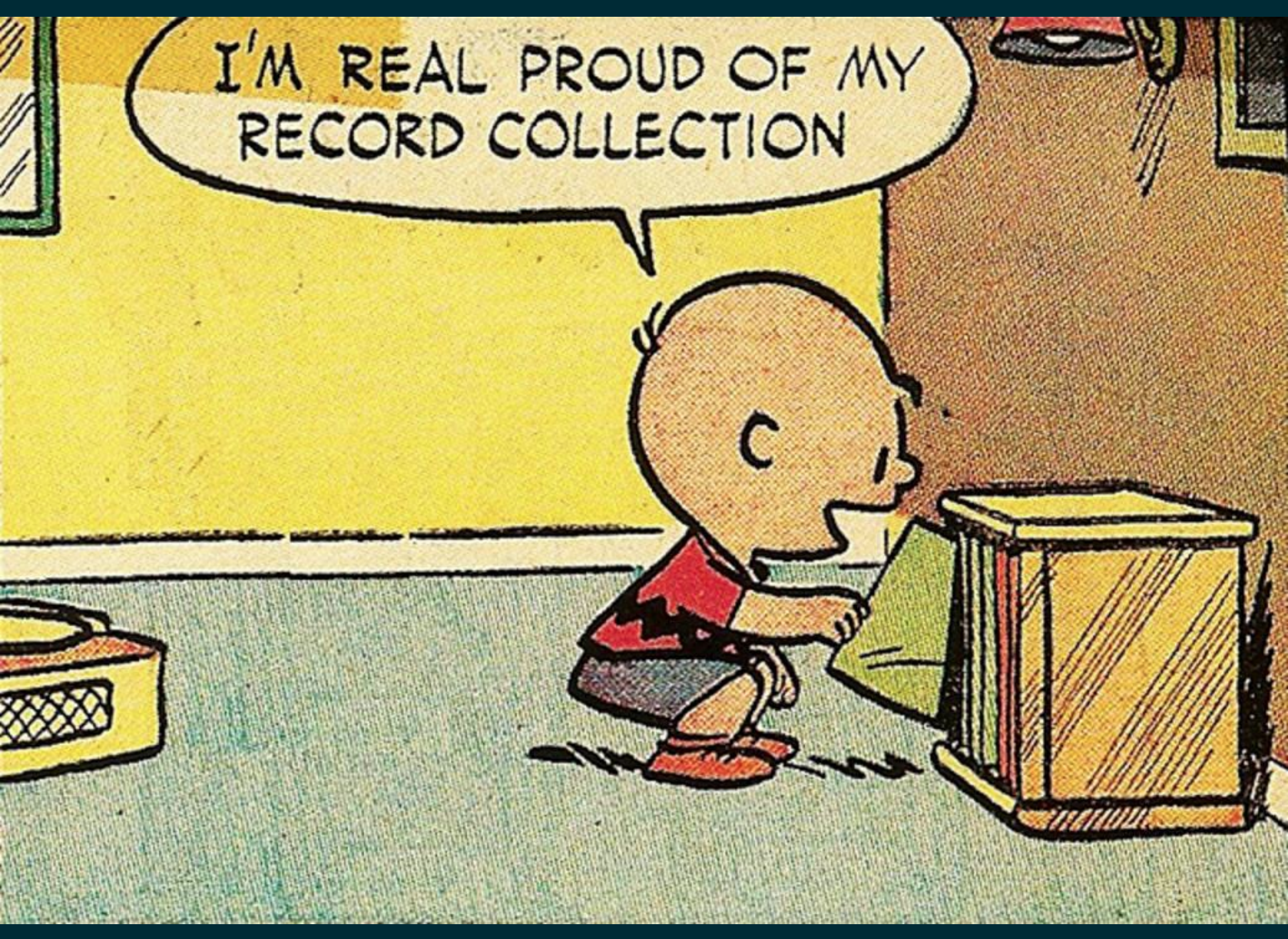
Real World Scenarios For Modern CFML

Nolan Erck

South of Shasta Consulting

About Me

- Software Consultant (southofshasta.com)
  - Software Development, Training, Design
- ColdFusion, C++, Java, jQuery, PHP, Angular, Android, SQL, etc...
- Manager, SacInteractive User Group
- Reformed Video Game Developer (Grim Fandango, SimPark, StarWars Rogue Squadron, etc).
- Music Junkie


I'M REAL PROUD OF MY RECORD COLLECTION

Code And Slides

github.com/nolanerck

modern-cfml-demos

Today's Agenda

- Look at new language features CFML
- And related newer tools, etc
- Discuss briefly what they do
- And give non "hello world" examples of how to use them

First real-world tip

Install CommandBox!

- Seriously
- If you learn nothing else, learn this
- Really is a game-changer
- You do *not* have to be on ColdBox to use it
- All the Node/npm fun from JavaScript for the CFML world!
- Packages, registry, CLI tools, server management
- Written mostly in CFML
- Easy to modify, contribute to, learn, update, etc

CommandBox...

- Command line CFML tools
- *Incredibly* easy to set up new environments
- Switch to *any* version of CFML easily
- REPL
- Easy scaffolding for projects
- Basically all the things Node developers brag about
- But for CFML!
- My "CommandBox vs Node" preso from yesterday

Second Real World Tip...

Install TestBox!

TestBox

- Can test *any* CFML app
- Not just ColdBox apps!
- The better your code, the easier to test
- But can be used to test old cfincludes, no-framework apps, anything!
- "TestBox for non-ColdBox Apps" preso

"box install testbox"...boom, done

Closures

- Not a CFML specific feature
- Common in modern JavaScript
- Run code in its own "context"
- What the heck does that mean?

Closures

- Example from CFDocs.org

```
function helloTranslator( String helloWord ) {
    return function( string name ) {
        return "#helloWord#, #name#";
    };
}
```

Testbox Closure Example

```
function run( testResults, testBox ){
    // all your suites go here.

    describe( title="Customer Admin Functionality", body=function(){

        it( 'should create a new customer object correctly', function() {

            // test code goes here

        });

        it( 'should delete an existing customer correctly', function() {

            // test code goes here

        });
```

Testbox Closure Example

```
describe( "Test UserLoginService", function(){

    UserLoginService = new com.services.UserLoginService();

    it( "should have a getLogin method",
        function() {
        expect( UserLoginService ).toHaveKey( "getLogin", "did NOT find the getLogin method" );
    });

    it( "should save user activity to a log file", function(){
        UserLoginService.login( "username", "password" );

        // now check to make sure the "activitylog.txt" got updated
        try
        {
            fileContents = fileRead( "activitylog.txt" ); // what if that file doesn't exist?
                                                          // this test will fail with an error!
        }
        catch( e )
        {
            WriteOutput( "file doesn't exist or is locked!" );
            abort;
        }
    });

    it( "should have a resetPassword method", function(){
        expect( UserLoginService ).toHaveKey( "resetPassword", "did NOT find the resetPassword method" );
    });
});
```

Closures are all over TestBox

- Testing code is always important
- You can use TestBox to test *any* CF code
- Not just for ColdBox projects
- Can test cfinclude, functions, custom tags, whatever
- Deson't need to be a ColdBox app
- But you *do* need closures

But that's just how TestBox does it, right?

Angular Closure Example

```
describe('Customer Admin Functionality', () => {
  let component: AboutComponent;
  let fixture: ComponentFixture<AboutComponent>;

  it('should create a new customer object correctly', () => {

    // test code goes here

  });

  it('should delete an existing customer correctly', () => {

    // test code goes here

  });
```

They also work in CF tags!

But please try not to do that. :)

Member Functions

- Really don't *do* anything different
- Instead of ArrayLen(arr), it's arr.length()
- The "thing" has the function built inside
- Similar to Java, Groovy, JavaScript, etc
- Code looks more like other OO languages

Member Functions Examples

```
aryUsers = [ "John", "Paul", "George", "Ringo" ];

WriteOutput( "We have #ArrayLen( aryUsers )# users in the system." );

WriteOutput( "We have #aryUsers.len()# users in the system." );
```

Member Functions Examples

```
aryUsers = [ "John", "Paul", "George", "Ringo" ];

// ArrayFirst() and ArrayLast()
WriteOutput( aryUsers.First() );
WriteOutput( aryUsers.Last() );
```

Member Functions On a Literal

```
WriteOutput( [ "John", "Paul", "George", "Ringo" ].len() );
```

Member Functions Examples

- Lots More too
- ArrayMin(), max(), etc
- Array Slicing
- XML Member Functions
- Numeric Member Functions

But which syntax is faster?

It doesn't matter

Okay so why use it?

Modernize...or die!

Elvis, Ternary, and Safe Navigation

- All essentially short-hands for various if() statements
- Elvis is ?:
- Ternary is test-case ? expression : expression
- Safe Navigation is structName?.keyThatMayNotExist

Elvis Operator

```
if( not IsDefined( "foo" ) )
{
    answer = "bar";
}

answer = foo ?: "bar";

WriteOutput( "answer is: " & answer );
```

Elvis Operator

```
if( not IsDefined( "form.shippingAddress" ) )
{
    billingAddress = "Same As Shipping Address";
}

billingAddress = form.shippingAddress ?: "Same As Shipping Address";
```

Ternary Operator

```
BillingAddress = "123 Main Street";
ShippingAddress = "";

if( Len( ShippingAddress ) gt 0 )
{
    AddressUsed = ShippingAddress;
}
else
{
    AddressUsed = BillingAddress;
}

AddressUsed = Len( ShippingAddress ) gt 0 ? ShippingAddress : BillingAddress;
```

Safe Navigation Operator

```
userinfo = {
    name = "Mick Ronson",
    address = "64 Abbey Road",
    city = "San Francisco",
    state = "CA"
};

WriteOutput( userinfo.name );
WriteOutput( userinfo.address );
WriteOutput( userinfo.city );
WriteOutput( userinfo.state );

// calls to an API for more user related info
userInfoWithShippingDetails = getCustomerShippingInfo( userinfo );

WriteOutput( userInfoWithShippingDetails?.orderShippedDate );

if( StrucKeyExists( userInfoWithShippingDetails, "" ) )
{
    WriteOutput( userInfoWithShippingDetails.orderShippedDate );
}
```

But which syntax is faster?

It doesn't matter

Okay so why use it?

Modernize...or die!

QueryExecute()

- Cleaner way to execute queries in CFScript
- Less of the ".net" style block of code
- Essentially one line of code
- QueryExecute(sql, params, options);

QueryExcute() Older Syntax

```
qMovie = new com.adobe.coldfusion.query();q.setDatasource("moviedsn");

qMovie.setSQL( "SELECT * FROM tMovies
                WHERE MovieID :ID" );

qMovie.addParam( name="ID", value="3", cfsqltype="cf_sql_int" );

rsltMovie = q.execute();
```

QueryExecute()

```
<cfquery name="qGetMovies">
    SELECT MovieId, Title, ReleaseYear
    FROM tMovies
</cfquery>

<cfset qGetMovies = QueryExecute( "SELECT MovieId, Title, ReleaseYear FROM tMovies",
                                  {},
                                  { datasource="dsn" } ) />

<cfscript>
    qGetMovies = QueryExecute( "SELECT MovieId, Title, ReleaseYear FROM tMovies",
                               {},
                               { datasource="dsn" } );
</cfscript>
```

QueryExecute()

```
qGetMovies = QueryExecute( "SELECT MovieId, Title, ReleaseYear FROM tMovies
                            WHERE MovieID = :ID",
                            { ID = 5 },
                            { datasource="Movies" } );
```

this.datasource in Application.cfc

this.datasource = "moviesdsn"

```
qGetMovies = QueryExecute( "SELECT MovieId, Title, ReleaseYear FROM tMovies
                            WHERE MovieID = :ID",
                            { ID = 123 } );
```

But which syntax is faster?

It doesn't matter

Okay so why use it?

Modernize...or die!

Functional Programming

- filter()
- map()
- reduce()
- Convert queries to arrays for better APIs, etc

map()

- Takes an array of things, runs the *closure* on each item
- Similar to a for() loop
- But closures have "context"

map() example

```
aryUsers = [ "john@beatles.com", "paul@beatles.com", "george@beatles.com", "ringo@beatles.com" ];

aryUsers.map( function( user )
{
    UserService.RegisterNewUserInSystem( user );
});
```

reduce() example

```
let aryFlavorNoteURLs = data[ 0 ].field_spirit_flavor.map( a => { return a.url; } );
this.flavor_notes = [];

// fire the "flavor note IDs" in order and reduce() them on the way back
// so we're guaranteed they show up in the correct sequence when rendered on the screen.
// Unfortunately there's no way to do this via Drupal so we need this
// crazy pile of promises and reduce() logic to get the data back correctly.
let result  = aryFlavorNoteURLs.reduce( ( accumulatorPromise, nextID ) =>
{
    return new Promise( ( resolve, reject ) =>
    {
        return accumulatorPromise.then(() =>
        {
            return this.contentSvc.getFlavorNote( nextID ).then( fn => {
                    let _cleanNote = this.sanitizer.bypassSecurityTrustHtml( fn );

                    this.flavor_notes.push( _cleanNote );

                    resolve( nextID );
            });
        });
    });
}, Promise.resolve());
```

But which syntax is faster?

built-in map(), reduce(), etc is pretty fast

built-in map(), reduce(),etc is pretty fast

- Built-in functions
- Can probably replace larger loops and custom logic in your legacy apps
- Code will look and behave similar to modern JavaScript stacks too
- Some ACF servers have perf hit with large map() or reduce() loops
- Patching to latest updates fix this

Okay so why use it?

Modernize...or die!

Pick and Choose

- Don't have to change the entire app at once
- Swap in a new language construct one at a time
- Use TestBox to ensure you don't break anything
- None of these constructs are *box specific!
- Use them in your legacy apps today!

Other Resources

- CFML Reference on helpx.adobe.com
- cfdocs.org
- "Head First Design Patterns" book
- South of Shasta on-site and remote training classes
- Talk to people at the conference!

Learning in 30 minutes a day

Questions? Comments?
Need consulting help?

- southofshasta.com
- nolan@southofshasta.com
- Twitter: @southofshasta
- Github: nolanerck

github.com/nolanerck/modern-cfml-demos

Thanks!