The Many Facets of CommandBox

Dan Card

About Me

- Been using CF since 2003
- Currently the CTO for getSpringboard.com
- Adjunct Faculty at UMass Lowell
- Taught HS, Undergrad, Adult Continuing Ed
- Married with a 2.5 year old son

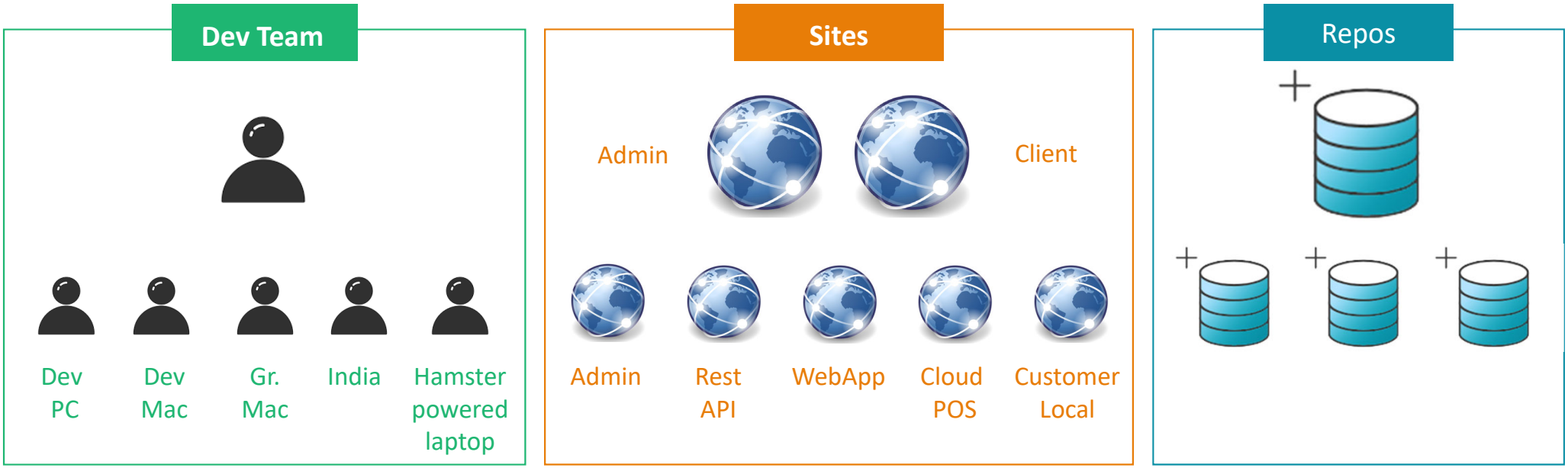
Presentation Disclaimer / Overview

- This is not a “deep dive” into CommandBox. It’s an overview of some of the techniques that exist with some code as an example.
- Had a choice of presenting the finished product or the “process” that went into it or a “problem → solution” format
- Went with “problem → solution“, mostly because we thought it made us look smarter than we felt while we were doing it.
- Evolved out of “absorbing” what was out there and then just trying it

- Github: https://github.com/djcard/ITBCBDemo

Situation Overview

- Springboard is a B2B company focusing on helping companies with the logistics of pickup and delivery laundry services



Steps and Goals

Steps

- Create Folder Structure
- Pull from repos

Goals

- Needs to mirror our production as exactly as possible
- The fewer clicks/steps/tools the better

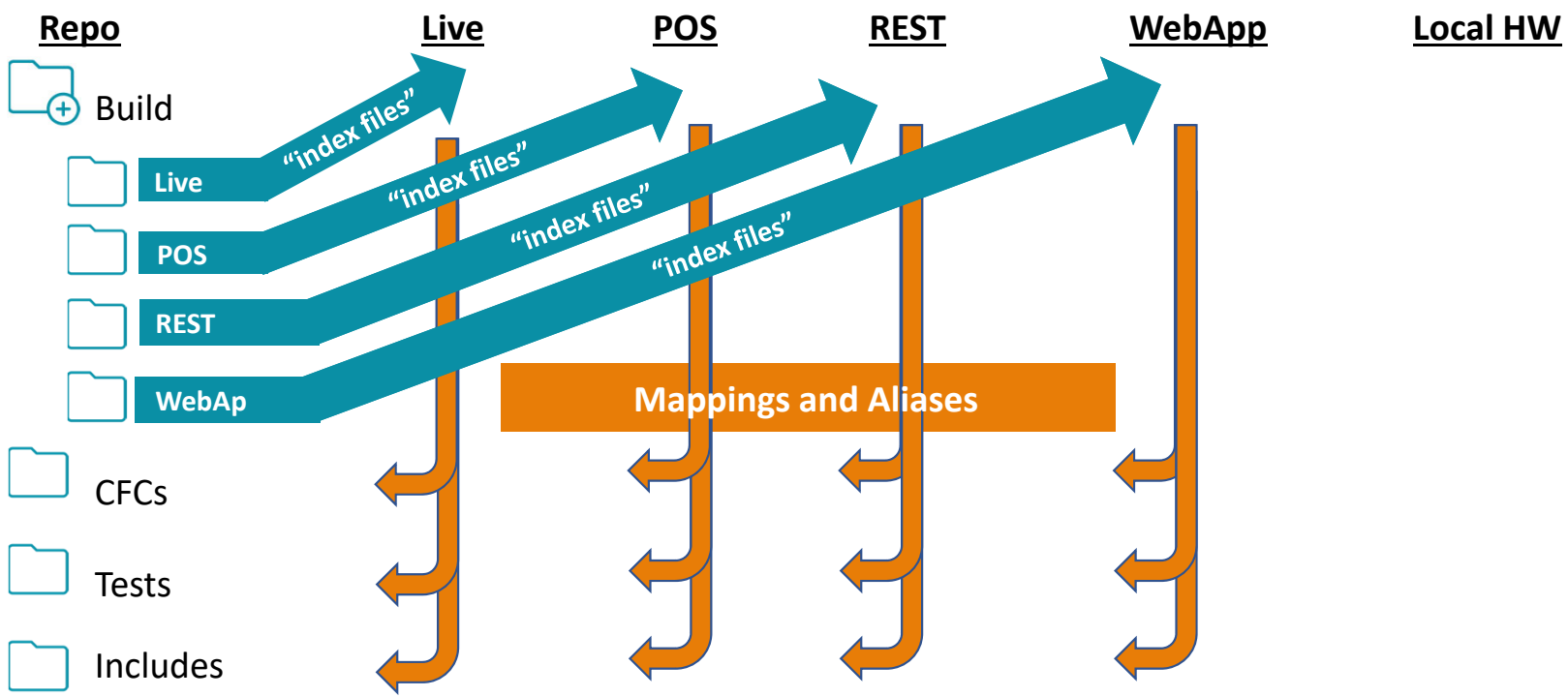
First We Need Some Folders and Files

- CommandBox can do file and folder manipulation
  - Mkdir *name* [true]
- Can run other programs on the command line with the "!" prefix
  - !git clone https://github.com/djcard/ITBFakerCoSites.git

*

COMMANDBOX

Folder Structure



Steps and Goals

Steps

- Create Folder Structure
- Pull from repos
- Move “index” files over

Goals

- Needs to mirror our production as exactly as possible
- The fewer clicks/steps/tools the better

*

COMMANDBOX

Challenge 1: 5 sites locally

- Simply Use Server Start
  - Starts up a Lucee Server (CFML)
  - Undertow (Web Server)
  - Uses 127.0.0.1 and a random port
  - Run as many servers as your resources will allow

*

COMMANDBOX

Challenge 2: Libraries and Dependencies

- Projects has several libraries and dependencies including
  - React / babel (js)
  - Bootstrap (css/js)
  - jQuery (js)
  - More
- How do we communicate to our team what is current and make it easy to install them?
- The box.json file describes a project and certain properties about it.

COMMANDBOX

Box.json

```
{
    "name":"Faker Co Main Site",
    "version":"0.0.1",
    "author":"Dan Card",
    "homepage":"http://fakerco.com/",
    "documentation":"Demo Project for ",
    "slug":"intotheboxfakerco",
    "shortDescription":"The FakerCo main site for Into the Box 2019.",
    "createPackageDirectory":false,
    "keywords":"Into The Box,CFML,Box,PHPStinks",
    "private":true,
    "contributors":[],
    "dependencies":{
        "app":"http://dev.getSpringboard.com/cbpackages/app.zip",
        "bootstrap4":"http://dev.getSpringboard.com/cbpackages/bootstrap4.zip",
    },
    "devDependencies":{},
    "installPaths":{
    "bootstrap4":"bower_components/bootstrap/",
    "jcardnci":"jars/",
        "popper.js":"bower_components/popper.js/"
    },
    "scripts":{},
    "ignore":[
        "**/.*",
        "test",
        "tests"
    ],
    "testbox":{}
}
```

Challenge 2: Libraries and Dependencies

- Projects has several libraries and dependencies including
  - React / babel (js)
  - Bootstrap (css/js)
  - jQuery (js)
  - More
- How do we communicate to our team what is current and make it easy to install them?
- The box.json file describes a project and certain properties about it.
- One of these is the dependencies (and dev dependencies). Typing “install” will start this process.

*

COMMANDBOX

Steps and Goals

Steps

- Create Folder Structure
- Pull from repos
- Move “index” files over
- Run “install” to install dependencies

Goals

- Needs to mirror our production as exactly as possible
- The fewer clicks/steps/tools the better

COMMANDBOX

Challenge 3: Specific Settings

Server	Special Considerations
Mainsite	ACF 2016 Custom Tag Datasource SSL Scheduled Tasks Mail Servers
NewProduct	ACF 2016 Datasource
RestServices	ACF 2016 Rest Framework Datasource FIXED URL AND PORT
WebApp	ACF2016
Local “site”	Extra Jar files FIXED URL AND PORT

Server.json

- JSON file
- Sits in the root of the server

  (not in production!)

- Establishes startup settings
  - CF
    - Enable Rest Service
  - Java
    - Jvm settings
  - Web
    - Directory Browsing
    - Default docs
    - Error Pages
  - Networking
    - Host (IP)
    - Port
    - SSL

COMMANDBOX

Main Site Server.json

```
{
"web":{
    "ssl":{
        "enable":true,
        "port":"443"
    },
    "http":{
        "port":"80"
    },
    "host":"fakerCo.local"
},
"app":{
    "cfengine":"adobe@2016.0.05+303689",
}
}
```

COMMANDBOX

New Product Server.json

```
{
    "app":{
        "cfengine":"adobe@2016"
    },
    "web":{
        "http":{
            "port":"80"
        },
        "host":”fakerCoNP.local"
    }
}
```

COMMANDBOX

RestServices Server.json

```
{
    "app":{
        "cfengine":"adobe@2016.0.05+303689",
        "restMappings":"/rest/*"
    },
    "web":{
        "http":{
            "port":"80"
        },
        "host":"restservices.fakerCo.local"
    }
}
```

COMMANDBOX

WebApp Server.json

```
{
  "app":{
    "cfengine":"adobe@2016"
  },
  "web":{
    "host":"fakerCoWebApp.local",
    "http":{
      "port":"80"
    }
  }
}
```

COMMANDBOX

Localized Server.json

```
{
  "app":{
    "cfengine":"adobe@2016",
    "libDirs":"jars"

  },
  "web":{
    "host":"fakerCoWebApp.local",
    "http":{
      "port":"80"
    }
  }
*}
```

COMMANDBOX

Challenge 3 : Specific Settings

Server	Special Considerations
Mainsite	ACF 2016 **Custom Tag** **Datasource** SSL **Scheduled Tasks** **Mail Servers**
NewProduct	ACF 2016 **Datasource**
RestServices	ACF 2016 Rest Framework **Datasource** **FIXED URL AND PORT** **Mappings**
WebApp	ACF2016
Local "site"	Extra Jar files FIXED URL AND PORT

CommandHost Updater / Our First Module

- CommandBox Module

What is a Module?

- Additional Functionality that you can download and install
- “install commandbox-hostupdater”

From where do you download and install these modules?

COMMANDBOX

Whence comes your code?

- Forgebox

What is Forgebox?

- "…the package repository and software directory for ColdFusion (CFML)"
  - CommandBox Modules
  - CF based Applications
  - Frameworks
  - Whatever you put up there
- When you type "install" in CommandBox, it's where it looks first

FORGEBOX

Whence comes your code?

- Forgebox
- Git / Github
- .ZIP files (http/s)
- Java
- S3
- CFLib
- RIAForge
- Jar
- Gist

What is CommandBox Host Updater

- CommandBox, on it's own, binds to an IP (typically 127.0.0.1:????)
- CommandBox Host Updater does two things
  - Adds internal facing IP addresses to your machine in the 127.127.0.1 to 127.127.255.255 range
  - Adds a host name to your host files that points to that IP address (this is why you need to run as an administrator)
- This is key because:
  - It gives predictability to your URLs
  - Many servers can use port 80 because they are all on different IP addresses
- install commandbox-hostupdater

*

Challenge 3: Specific Settings

Server	Special Considerations
Mainsite	ACF 2016 Custom Tag Datasource SSL Scheduled Tasks Mail Servers Mappings
NewProduct	ACF 2016 Datasource Mappings
RestServices	ACF 2016 Rest Framework Datasource FIXED URL AND PORT Mappings
WebApp	ACF2016
Local “site”	Extra Jar files FIXED URL AND PORT

More Settings Than server.json!

- Need to mirror your production server as exactly as possible.
- There are over 180 individual config items.
- CFConfig is a module for CommandBox which can help manage them.

CFCONFIG

What is CFConfig?

CFConfig gives you the ability to manage most every setting that shows up in the web administrator, but instead of logging into a web interface, you can mange it from the command line by hand or as part of a scripted server setup. You can seamlessly transfer config for all the following:

- CF Mappings
- Datasources
- Mail servers
- Request, session, or application timeouts
- Licensing information (for Adobe)
- Passwords
- Template caching settings
- Basically any settings in the web based administrator

CFCONFIG

More Settings Than server.json!

- Need to mirror your production server as exactly as possible.
- There are ***over 180*** individual config items.
- CFConfig is a module for CommandBox which can help manage them.
- Can edit them these settings from CommandLine (CommandBox)
- Move them from version to version (testing upgrades)
- You can do an exact clone or exact “translation” of your server into a different version or even a different engine.
- install commandbox-cfconfig

Main Site Server.json with CFConfig Settings

```
"web":{
    "ssl":{
        "enable":true,
        "port":"443"
    },
    "http":{
        "port":"80"
    },
    "host":"getSpringboard.local"
},
"app":{
    "cfengine":"adobe@2016.0.05+303689"
},
*"cfconfigfile":"TestSiteSettingsLive.json"
```

COMMANDBOX

Our site now has all settings from production

- Including:
  - It's admin password
  - All scheduled tasks in the same state that they were in production (running?!)
  - All datasources pointing to live databases
  - All custom tags and mappings pointing to the file path on the production server (which might not lineup)
  - All mail servers

- Fine if you're testing or doing trials, not good if you're setting up your dev environment or need to keep those passwords secure.

COMMANDBOX

Main Site Server.json with CFConfig Settings

```
"web":{
    "ssl":{
        "enable":true,
        "port":"443"
    },
    "http":{
        "port":"80"
    },
    "host":"getSpringboard.local"
},
"app":{
    "cfengine":"adobe@2016.0.05+303689",
    "libDirs":"jars"
},
"cfconfigfile":"TestSiteSettingsLive.json",
"CFConfigPauseTasks":"true"
```

COMMANDBOX

Steps and Goals

Steps

- Create Folder Structure
- Pull from repos
- Move “index” files over
- Run “install” to install dependencies
- Start/stop Servers
- Import settings
- Specific Settings

Goals

- Needs to mirror our production as exactly as possible
- The fewer clicks/steps/tools the better
- Bare minimum of effort to get edited files into the repo

CFCONFIG

After all: coders and techies are known for reading documentation and following directions, right?

Challenge 4: Automating all of this

- Need to change passwords to something non – production and (hopefully) memorable
- Datasources need to point to local databases
- Mappings (and aliases) need to point to the local environment
- Mail servers need to be turned off
- Schemas in the DB need to be updated
- If it’s too hard, people won’t use it and all of this is for nothing

Different Ways of Automating in CB

- Recipes
- Tasks
- Custom Module

Creating A Custom Module In CommandBox

- Three Required Elements
  - ModuleConfig.cfc
  - Namespace
  - A CFC with a function called “run”

Technique 1: Config Settings and “Asking”

- We need to know where the folders are for this setup
- We need the desired passwords
- We need to know the info for the datasources (location and creds)
- Config Settings are one way of storing info for your modules
- Create an intro to the process and some “call and response” with:
  - Print.line();
  - Confirm();
  - Ask();

*

CFCONFIG

Steps and Goals

Steps

- ~~Gather info~~
- ~~Create Folder Structure~~
- Pull from repos
- Move “index” files over
- Run “install” to install dependencies
- Start/stop Servers
- Import settings
- Specific Settings

Goals

- Needs to mirror our production as exactly as possible.
- The fewer clicks/steps/tools the better.
- Bare minimum of effort to get edited files into the repo.

Technique 2: Using “Command” and “!”

- Command is how to send…well….commands to the CB shell.
- The simplest is command().run();
- Almost (?) anything you can run on the CLI you can run from within a module.

COMMANDBOX

Steps and Goals

Steps

- ~~Create Folder Structure~~
- ~~Pull from repos~~
- ~~Move “index” files over~~
- Run “install” to install dependencies
- Start/stop Servers
- Import settings
- Specific Settings

*

Goals

- Needs to mirror our production as exactly as possible
- The fewer clicks/steps/tools the better
- Bare minimum of effort to get edited files into the repo

Technique 3: SubCommands

- Small pieces in your namespace
- Allow individual steps to be run on their own

*

COMMANDBOX

Steps and Goals

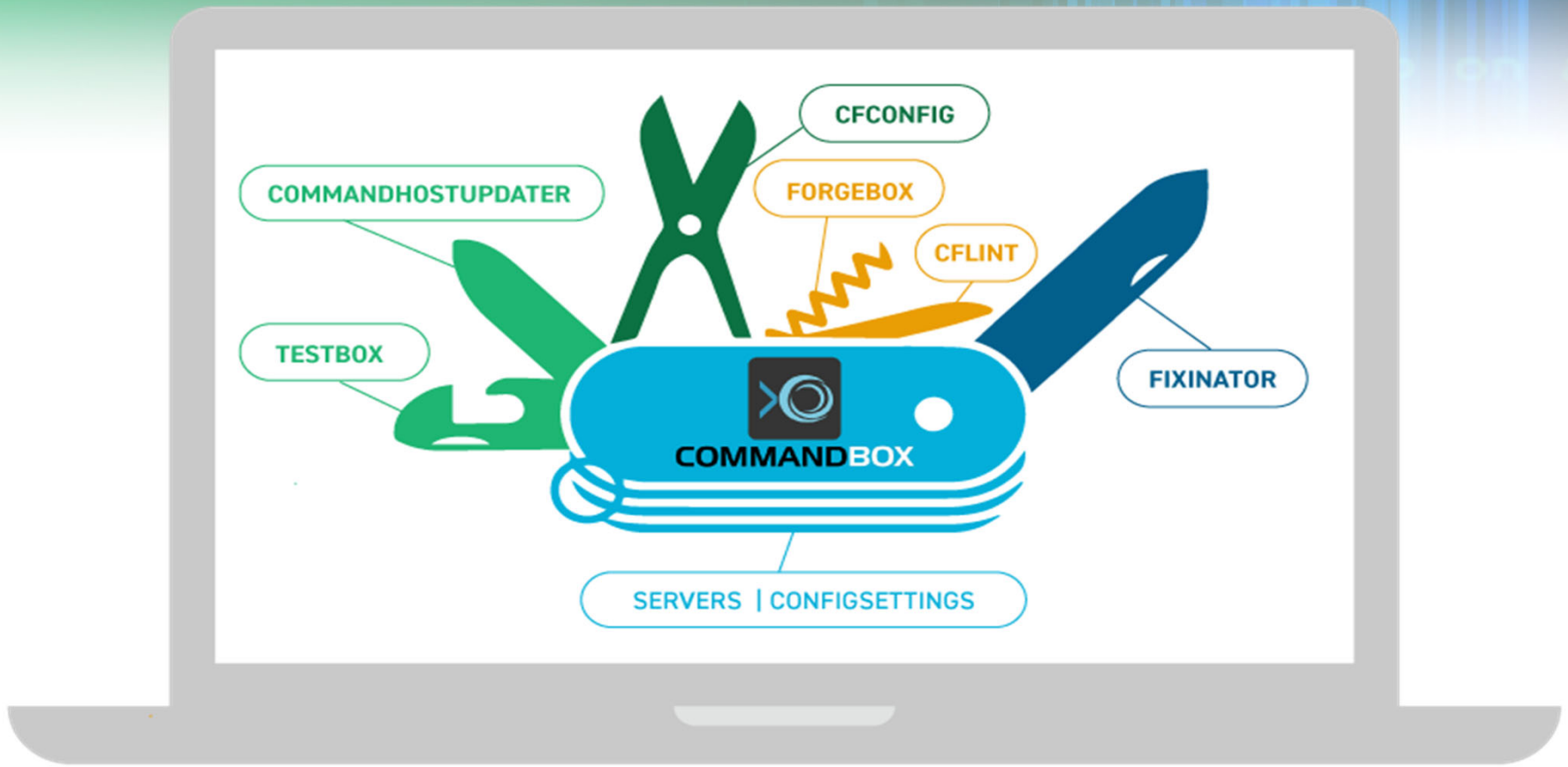
Steps

- ~~Create Folder Structure~~
- ~~Pull from repos~~
- ~~Move “index” files over~~
- ~~Run “install” to install dependencies~~
- ~~Start/stop Servers~~
- ~~Import settings~~
- ~~Specific Settings~~

Goals

- Needs to mirror our production as exactly as possible
- The fewer clicks/steps/tools the better
- Bare minimum of effort to get edited files into the repo

CommandBox is a Growing EcoSystem



Short list of tools (there are many many more!)

Module	Function
cfscriptme-command	Converts tag based CFML to script syntax
CFConfig	Allows Configuration and Transfer of Settings between CFML servers
CommandBox Host Updater	Allow host names, not just IPs for CB Servers
CFLint	Checks CML Syntax and best practices
Fixinator	Scans CFML code for security red flags
Fusion Reactor	Puts Fusion Reactor module into your site

Questions? Thoughts?