
ColdBox

ColdFusion Powered

Mobile Applications

CFCamp 2011 - Munich

ColdBox

Who am I?

Who am I?

- Luis Majano - Computer Engineer
- Born in El Salvador ---------->

- President of Ortus Solutions
- Manager of the IECFUG (www.iecfug.com)
- Creator of ColdBox, MockBox, LogBox, CacheBox, WireBox, ContentBox, or anything Box!
- Documentation Weirdo!

Ortus Solutions
A Software Revolution

Professional Open Source

Professional Open Source

- **ColdBox Platform, Relax, ContentBox, CodexWiki, etc**
- Professional Training & Books
- Support & Mentoring Plans
- Architecture & Design Sessions
- Server Tuning & Setup
- Code Reviews & Sanity Checks
- Consulting
- We can teach salsa dancing too!

Ortus Solutions
A Software Revolution

www.ortussolutions.com
consulting@ortussolutions.com

What we will cover?

- Communication protocols & data formats
- Why power a mobile app via ColdFusion
- ColdBox RESTful web services
- Resources identification & domain analysis
- RESTful Modeling via Relax!
- Application development basics
- Documentation, yes Documentation!
- Cool Demo

Getting Started

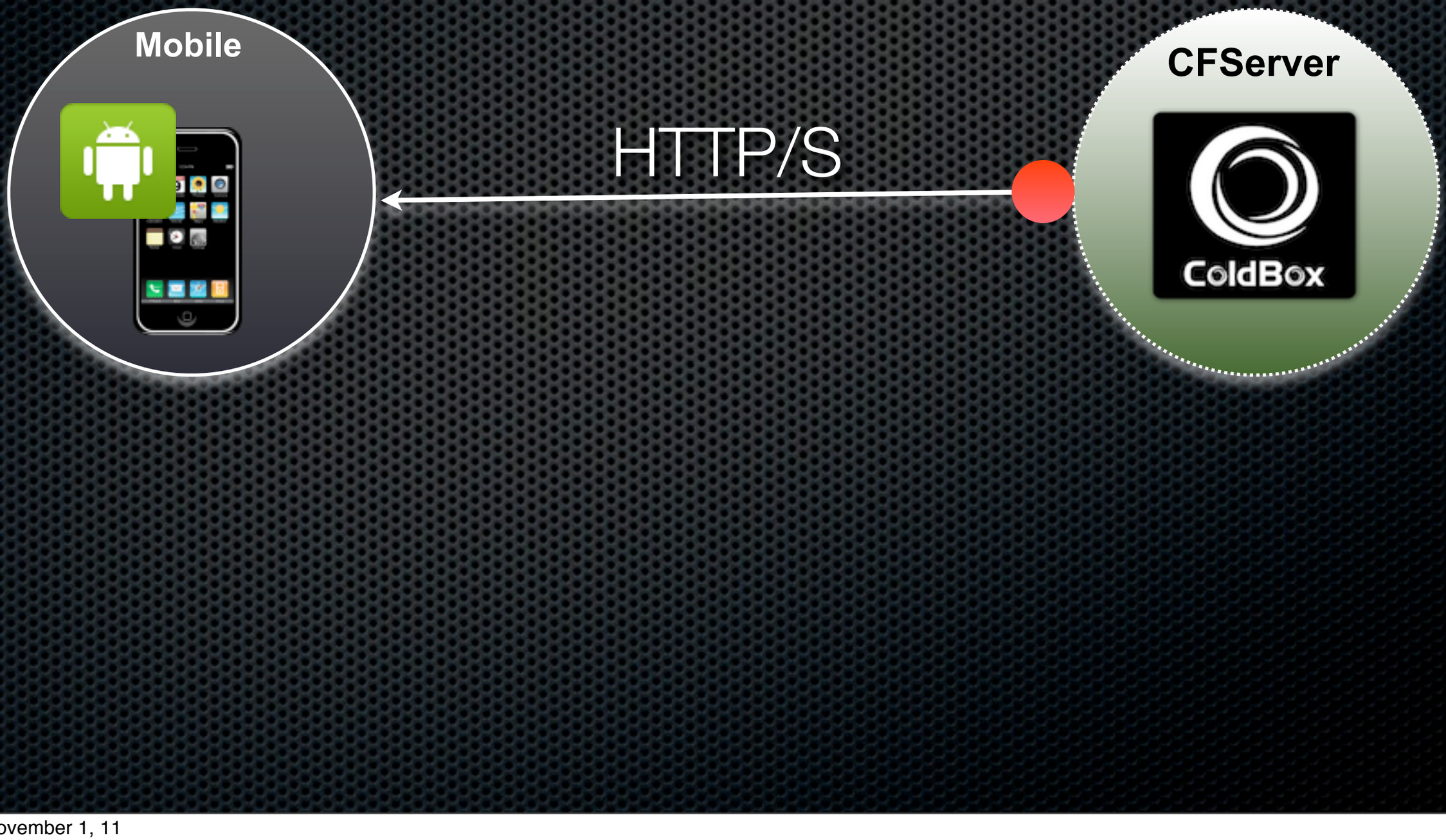
Getting Started

Getting Started

Getting Started

Getting Started

Getting Started



Getting Started

Communication Protocols

Communication Protocols

- HTTP/S is our main protocol of communication
  - **RESTful web services or**
  - SOAP
  - XML-RPC
- Data Formats
  - XML
  - **JSON**

Our Recommendations

Our Recommendations

- **RESTful** web services
  - low ceremony services
  - resource oriented & not RPC method oriented
  - readability & abstractions
  - easy to refactor & expand
  - > bandwidth
- **JSON** over XML
  - less verbose
  - lightweight communication
  - better iPhone support

Representational State Transfer (REST)

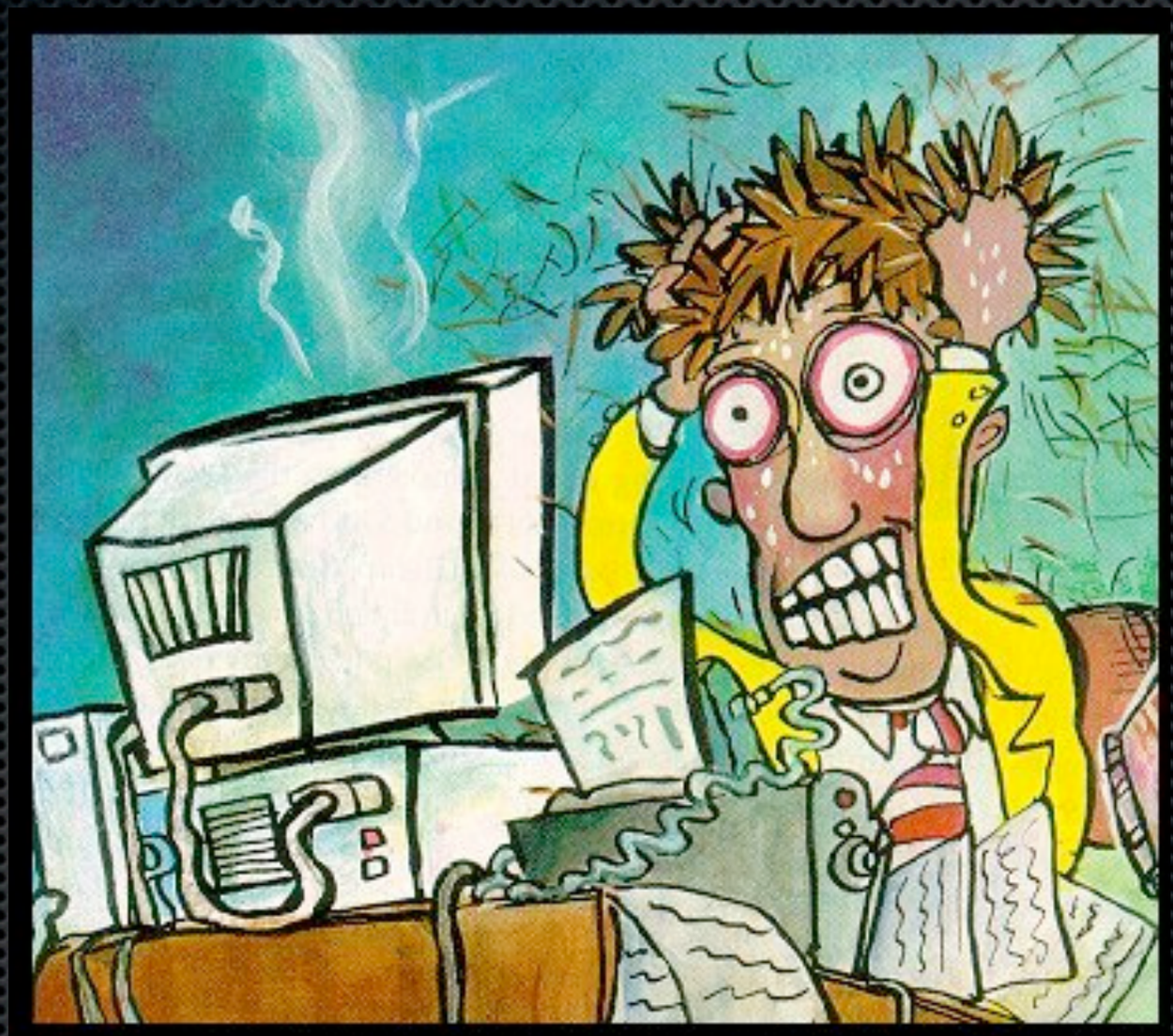
Representational State Transfer (REST)

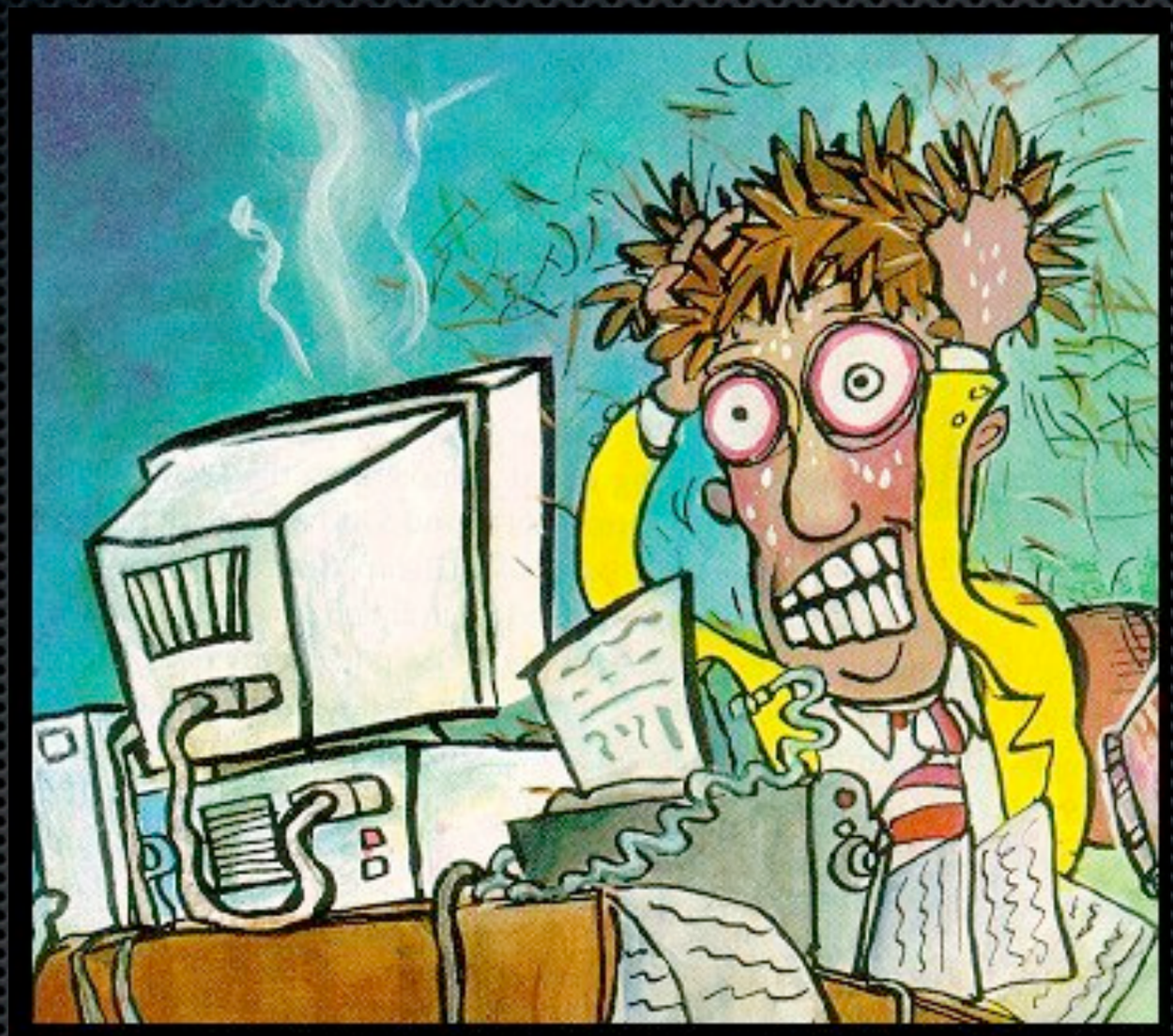
- Identify Resources to an URI
- The resource responds to different HTTP verbs
- The deeper you go in the resource -> More detail
- Use URL params + HTTP Headers
- Basic Verbs
  - GET - Represent resource(s)
  - PUT - Update
  - POST - Save
  - DELETE - Delete

```
/resource
/resource/detail
/resource/detail/moredetail?param1=value
```

What powers our Mobile Devices

What do I do?





ColdFusion+ColdBox

Testing

Modularity

Extensibility

Utilities

SES URLs

Logging

Scalability

Caching

Debuggers

Data Formats

HTTP Security

HTTP Verb Recognition

Why ColdBox?

Why ColdBox?

- Built in SES and RESTFul resource management

Why ColdBox?

- Built in SES and RESTFul resource management
- HTTP verb recognitions + Routing (POST,GET,PUT,DELETE,HEAD,OPTIONS)

Why ColdBox?

- Built in SES and RESTFul resource management
- HTTP verb recognitions + Routing (POST,GET,PUT,DELETE,HEAD,OPTIONS)
- Data/ORM transformations to XML, JSON, WDDX, Custom

Why ColdBox?

- Built in SES and RESTFul resource management
- HTTP verb recognitions + Routing (POST,GET,PUT,DELETE,HEAD,OPTIONS)
- Data/ORM transformations to XML, JSON, WDDX, Custom
- RESTful resource security and Basic Authentication

Why ColdBox?

- Built in SES and RESTFul resource management
- HTTP verb recognitions + Routing (POST,GET,PUT,DELETE,HEAD,OPTIONS)
- Data/ORM transformations to XML, JSON, WDDX, Custom
- RESTful resource security and Basic Authentication
- Integration testing and mocking facilities

Why ColdBox?

- Built in SES and RESTFul resource management
- HTTP verb recognitions + Routing (POST,GET,PUT,DELETE,HEAD,OPTIONS)
- Data/ORM transformations to XML, JSON, WDDX, Custom
- RESTful resource security and Basic Authentication
- Integration testing and mocking facilities
- Logging, profiling and tuning

Why ColdBox?

- Built in SES and RESTFul resource management
- HTTP verb recognitions + Routing (POST,GET,PUT,DELETE,HEAD,OPTIONS)
- Data/ORM transformations to XML, JSON, WDDX, Custom
- RESTful resource security and Basic Authentication
- Integration testing and mocking facilities
- Logging, profiling and tuning
- Resource caching and scalability

Why ColdBox?

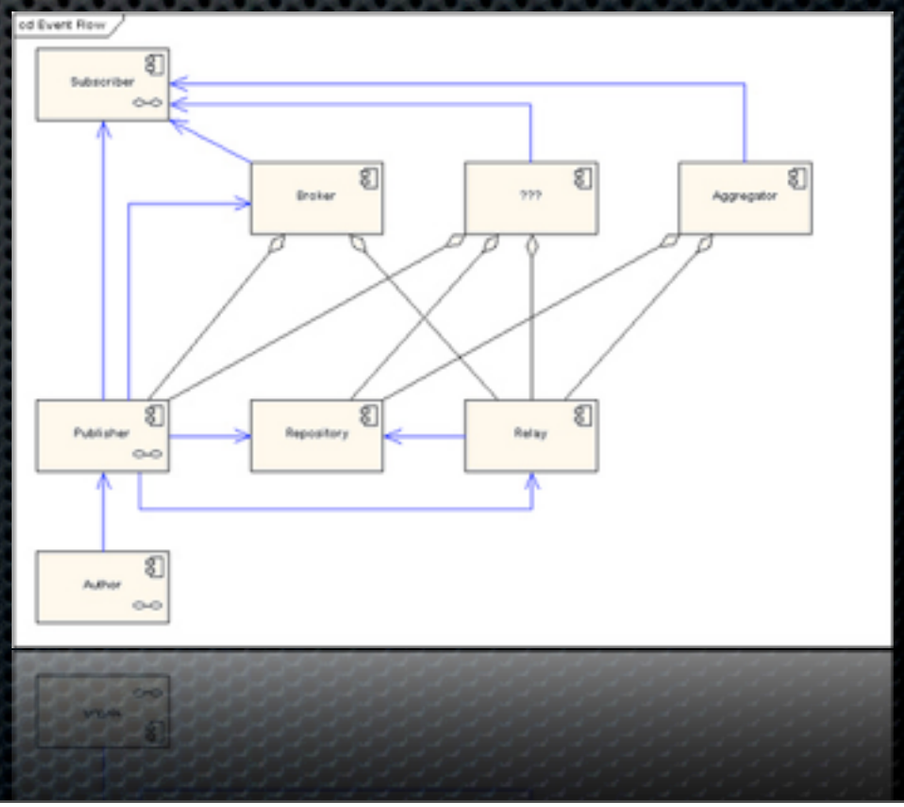
- Built in SES and RESTFul resource management
- HTTP verb recognitions + Routing (POST,GET,PUT,DELETE,HEAD,OPTIONS)
- Data/ORM transformations to XML, JSON, WDDX, Custom
- RESTful resource security and Basic Authentication
- Integration testing and mocking facilities
- Logging, profiling and tuning
- Resource caching and scalability

“Everything you need to power a mobile app!”

Planning our Service

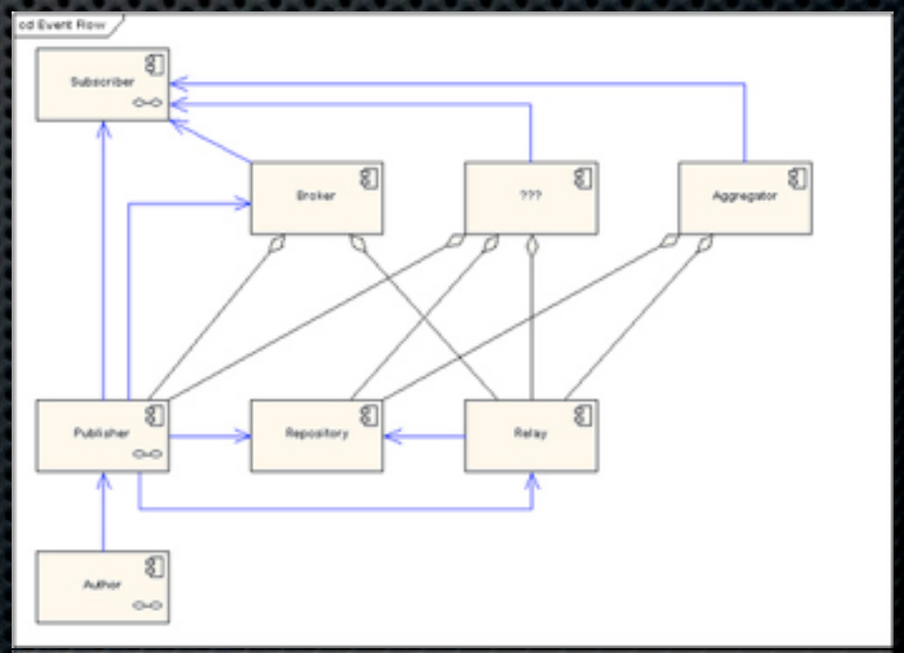
Planning our Service

- Domain Analysis & Design (UML)



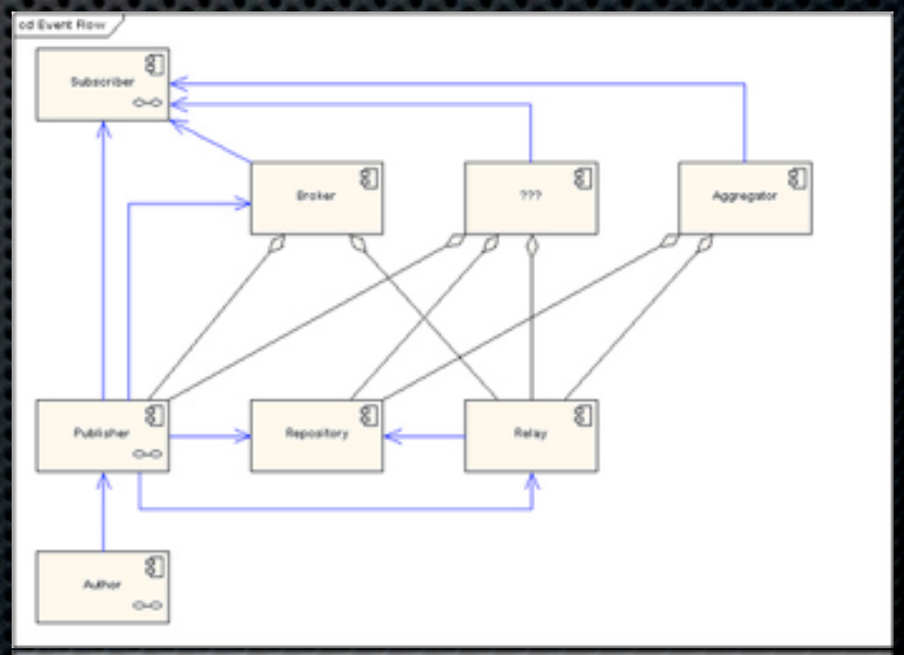
Planning our Service

- Domain Analysis & Design (UML)
- Plan our model layer, build it and test it, yes TEST IT!



Planning our Service

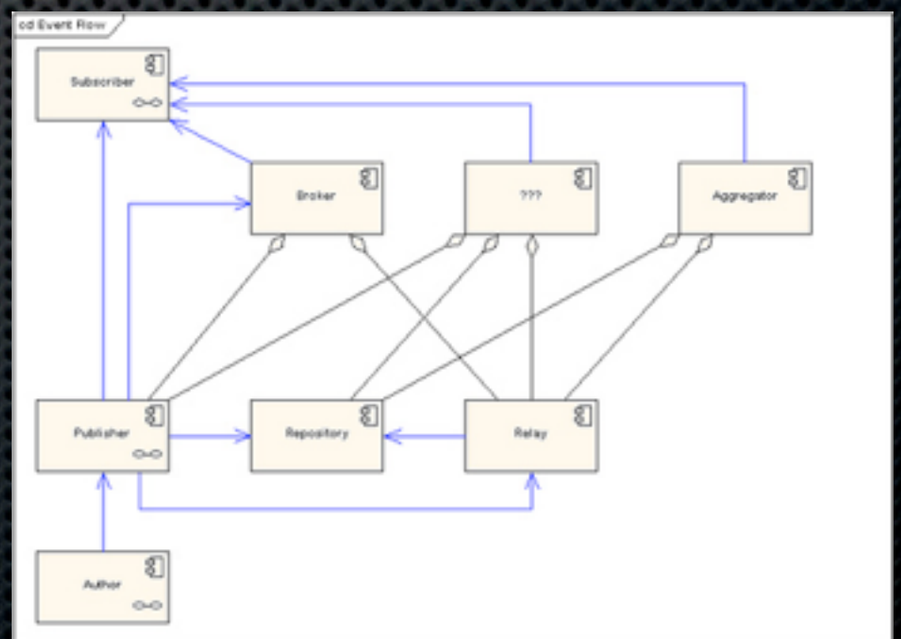
- Domain Analysis & Design (UML)
- Plan our model layer, build it and test it, yes TEST IT!
- Think about API Security (If Any)



Planning our Service

- Domain Analysis & Design (UML)
- Plan our model layer, build it and test it, yes TEST IT!
- Think about API Security (If Any)
- **Resource Identification & Planning**

```
/api/user/:username?
 - GET -> index
 - POST -> create
 - PUT -> update
 - DELETE -> remove
```



API Security

API Security

- Approaches:
  - Registered Client API Keys or Tokens
  - API Key + Secret Encryption Keys (Like Amazon)
  - Basic Authentication (At least its something!)
  - IP Based Filtering/Tagging (Programmatic/Firewall/Etc)
  - Headers
- HTTPS is a must!
- HTTP Method Security
- Request Throttling
- How secure do you need to be?

Let’s build our service!

now?

Right Tools

Right Tools

- CFBuilder + ColdBox Platform Utilities
  - Code Generation
  - Templating
  - Syntax Highlighting
  - Code Introspection
  - API Quick Docs
  - Much more..

Right Tools

Right Tools

- HTTP Monitors/Proxies
  - Charles, FireBug, HTTPFox
- RESTFul Consoles
  - Relax
  - Eclipse
  - REST Client for FireFox

Right Tools

Right Tools

- **RESTFul Tools For Lazy Experts**
- ColdBox Module
- RESTful API resource modeling
- Documentation Generation
  - CodexWiki, Mediawiki, Trac, PDF, HTML
- Service Logging & Monitoring
- RelaxURL Console (Pronounced “Relaxer”)
  - Test your resources
  - Save your requests for later testing

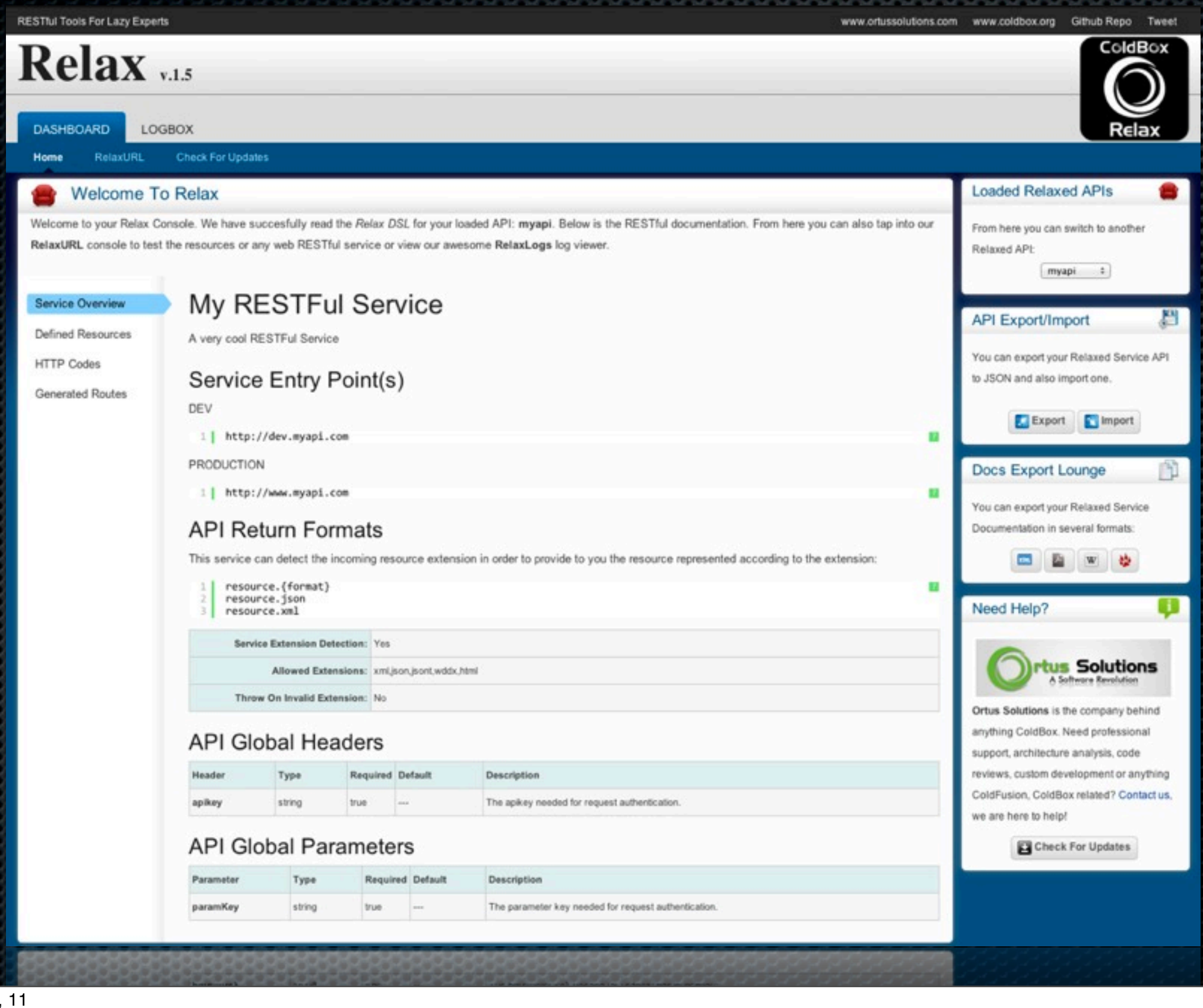
Service Setup

Service Setup

- Generate application
- Configure Logging (LogBox)
  - DB Appender 4 Relax Monitoring
- Configure 4 Full URL Rewrites
  - mod_rewrite, ISAPI, etc
- Install Relax Module
- Create our own Relaxed API


ColdBox
Relax

Let's Start This Bad Boy Up!



Relax Configuration

```
// Relax Configuration Settings
settings = {
    // Relax Version: DO NOT ALTER
    version = this.version,
    // The location of the relaxed APIs, in instantiation path
    apiLocationPath = "#moduleMapping#.resources",
    // Default API to load
    defaultAPI = "myapi",
    // History stack size, the number of history items to track in the RelaxURL
    maxHistory = 10,
    // logbox integration information needed for log viewer to work
    // this means that it can read tables that are written using the logbox's DB Appender.
    relaxLogs = {
        // THE CF DATASOURCE NAME
        datasource = "relax",
        // THE DB TO USE FOR LOGS, AVAILABLE ADAPTERS ARE: mysql, mssql, postgres, oracle
        adapter = "mysql",
        // THE TABLE WHERE THE LOGS ARE
        table    = "api_logs",
        // PAGING MAX ROWS
        maxRows = 50,
        // PAGING CARROUSEL BANDGAP
        bandGap = 3
    }
};
```

relax/ModuleConfig.cfc

Relax Configuration

```
// Relax Configuration Settings
settings = {
    // Relax Version: DO NOT ALTER
    version = this.version,
    // The location of the relaxed APIs, in instantiation path
    apiLocationPath = "#moduleMapping#.resources",
    // Default API to load
    defaultAPI = "myapi",
    // History stack size, the number of history items to track in the RelaxURL
    maxHistory = 10,
    // logbox integration information needed for log viewer to work
    // this means that it can read tables that are written using the logbox's DB Appender.
    relaxLogs = {
        // THE CF DATASOURCE NAME
        datasource = "relax",
        // THE DB TO USE FOR LOGS, AVAILABLE ADAPTERS ARE: mysql, mssql, postgres, oracle
        adapter = "mysql",
        // THE TABLE WHERE THE LOGS ARE
        table    = "api_logs",
        // PAGING MAX ROWS
        maxRows = 50,
        // PAGING CARROUSEL BANDGAP
        bandGap = 3
    }
};
```

relax/ModuleConfig.cfc

Relax Configuration

```
// Relax Configuration Settings
settings = {
    // Relax Version: DO NOT ALTER
    version = this.version,
    // The location of the relaxed APIs, in instantiation path
    apiLocationPath = "#moduleMapping#.resources",
    // Default API to load
    defaultAPI = "myapi",
    // History stack size, the number of history items to track in the RelaxURL
    maxHistory = 10,
    // logbox integration information needed for log viewer to work
    // this means that it can read tables that are written using the logbox's DB Appender.
    relaxLogs = {
        // THE CF DATASOURCE NAME
        datasource = "relax",
        // THE DB TO USE FOR LOGS, AVAILABLE ADAPTERS ARE: mysql, mssql, postgres, oracle
        adapter = "mysql",
        // THE TABLE WHERE THE LOGS ARE
        table    = "api_logs",
        // PAGING MAX ROWS
        maxRows = 50,
        // PAGING CARROUSEL BANDGAP
        bandGap = 3
    }
};
```

relax/ModuleConfig.cfc

Relax Configuration

```
// Relax Configuration Settings
settings = {
    // Relax Version: DO NOT ALTER
    version = this.version,
    // The location of the relaxed APIs, in instantiation path
    apiLocationPath = "#moduleMapping#.resources",
    // Default API to load
    defaultAPI = "myapi",
    // History stack size, the number of history items to track in the RelaxURL
    maxHistory = 10,
    // logbox integration information needed for log viewer to work
    // this means that it can read tables that are written using the logbox's DB Appender.
    relaxLogs = {
        // THE CF DATASOURCE NAME
        datasource = "relax",
        // THE DB TO USE FOR LOGS, AVAILABLE ADAPTERS ARE: mysql, mssql, postgres, oracle
        adapter = "mysql",
        // THE TABLE WHERE THE LOGS ARE
        table    = "api_logs",
        // PAGING MAX ROWS
        maxRows = 50,
        // PAGING CARROUSEL BANDGAP
        bandGap = 3
    }
};
```

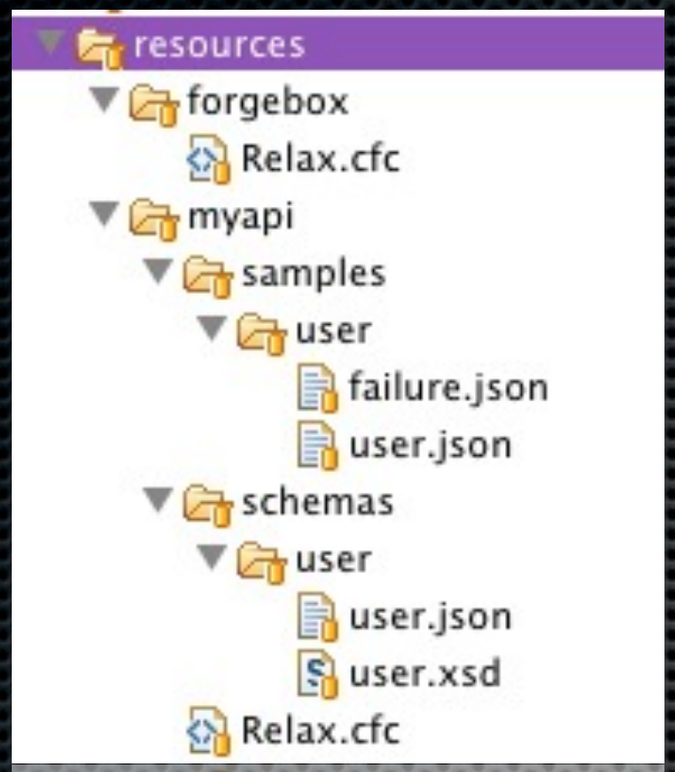
relax/ModuleConfig.cfc

Relax Configuration

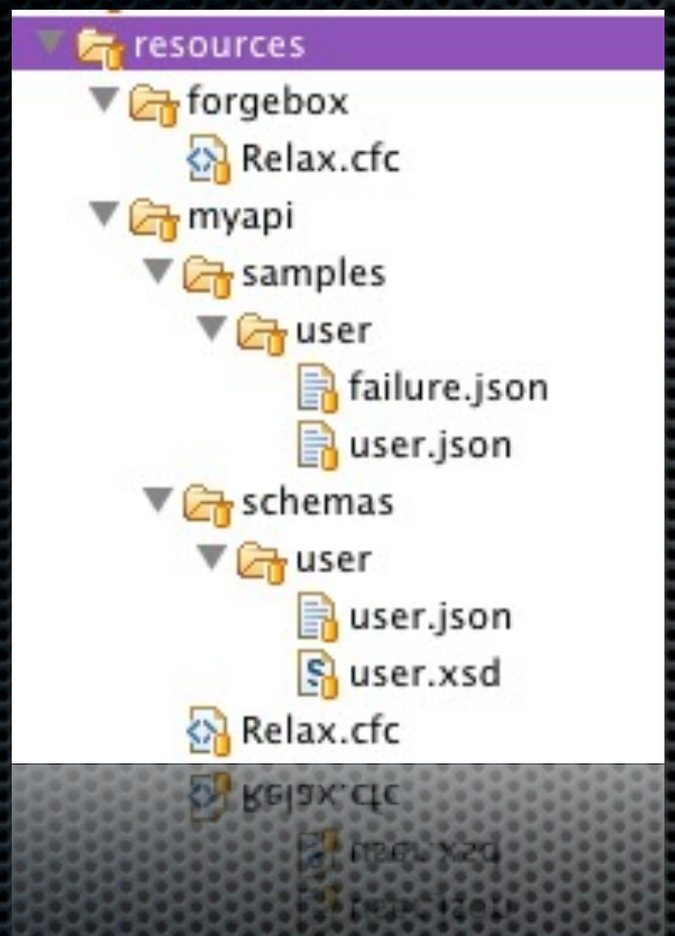
```
// Relax Configuration Settings
settings = {
    // Relax Version: DO NOT ALTER
    version = this.version,
    // The location of the relaxed APIs, in instantiation path
    apiLocationPath = "#moduleMapping#.resources",
    // Default API to load
    defaultAPI = "myapi",
    // History stack size, the number of history items to track in the RelaxURL
    maxHistory = 10,
    // logbox integration information needed for log viewer to work
    // this means that it can read tables that are written using the logbox's DB Appender.
    relaxLogs = {
        // THE CF DATASOURCE NAME
        datasource = "relax",
        // THE DB TO USE FOR LOGS, AVAILABLE ADAPTERS ARE: mysql, mssql, postgres, oracle
        adapter = "mysql",
        // THE TABLE WHERE THE LOGS ARE
        table    = "api_logs",
        // PAGING MAX ROWS
        maxRows = 50,
        // PAGING CARROUSEL BANDGAP
        bandGap = 3
    }
};
```

relax/ModuleConfig.cfc

Prepare API Resources



Prepare API Resources



- Create directory structure to model your API
- MyAPI
  - Relax.cfc
    - Describes & Models Your RESTful service
  - Samples/Schemas (optional)

Relax Modeling DSL

Relax Modeling DSL

- Create a simple CFC
- Create a set of public properties (Implicit structs and arrays)

Relax Modeling DSL

- Create a simple CFC
- Create a set of public properties (Implicit structs and arrays)

Key	Description
this.relax	Service title, description, tiered entry points, extension detection, valid format extensions, etc
this.globalHeaders	Define service global headers: [name,description,required,default,type]
this.globalParameters	Definite service global parameters: [name,description,required,default,type]
this.resources	A collection of resources that identify your service

Resource Modeling DSL

Resource Modeling DSL

Resource Keys	Description
pattern	The SES pattern also used for coldbox route generation
handler	The handler that responds to the pattern used for route generation
action	Simple or RESTful action used for route generation
placeholders	Collection that describes the pattern placeholders: [name,description,type,required,default]
description	Resource description, can use HTML
methods	HTTP methods implemented by this resource
headers	A collection of headers this resource responds to
parameters	A collection of parameters this resource responds to
response	Holds more metadata about resource response schemas and samples
response.schemas	A collection of different schema representations: [format,description,body]
response.samples	A collection of different sample representations: [format,description,body]

RESTful Routing

RESTful Routing

```
addRoute(pattern="API/settings",  handler="General",action="settings");
addRoute(pattern="API/echo/:name?",handler="General",action="{'POST':'echo'}");
addRoute(pattern="API/user/:username?",
        handler="admin:rest.user",
        action="{'GET':'index',‘PUT’:‘update’,‘POST’:‘add’,‘DELETE’:‘remove’}");
```

- **Pattern** = The URI of the resource (Dynamic)
- **Handler** = The controller to handle the event
- **Action** = Implicit Struct or JSON Struct
  - Maps HTTP verbs to handler methods

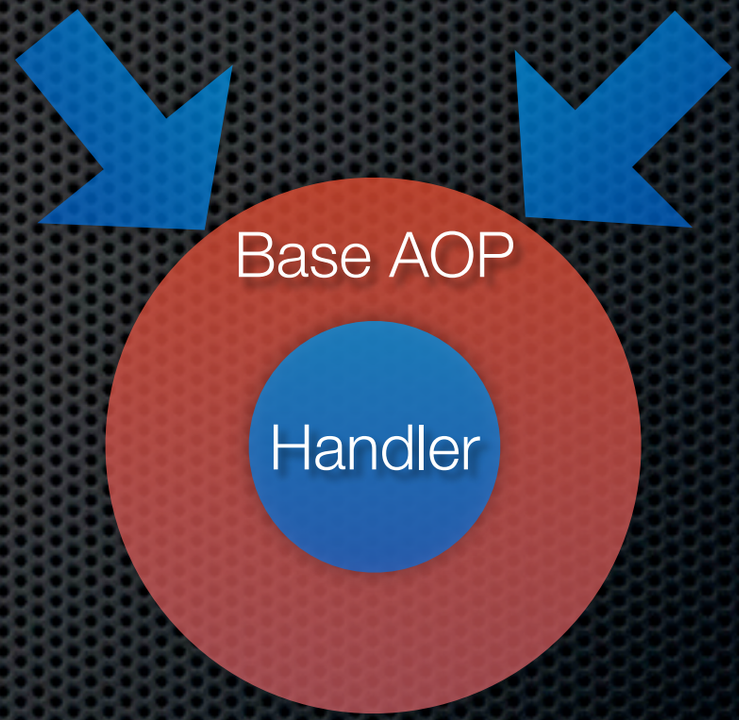
RESTful Handlers

RESTful Handlers

- Create a base REST handler
  - Data uniformity
  - Error uniformity
  - All via ColdBox AOP
- Create our RESTful Event Handlers

RESTful Handlers

- Create a base REST handler
  - Data uniformity
  - Error uniformity
  - All via ColdBox AOP
- Create our RESTful Event Handlers



Base Handler

Base Handler

```
component{

    function aroundHandler(event,targetAction){
        var prc = event.getCollection(private=true);
        log.debug("Request for route: #event.getCurrentRoute()#",getHTTPRequestData());
        prc.results = prepareResults();

        arguments.targetAction(event,rc,prc);

        event.renderData(data=prc.results,type="json");
    }
    private function prepareResults(){
        var results = {
            error      = false,
            messages   = "",
            data       = ""
        };
        return results;
    }
    function onError(event,faultAction,exception){
        var rc = event.getCollection();
        rc.results.error   = true;
        rc.results.messages = "The action:#faultaction# failed:
                               #exception.detail# #exception.message#";

        log.error(rc.results.messages, exception);

        // Headers
        event.setHTTPHeader(500,rc.results.messages)
           .renderData(data=rc.results,type="json");
    }
}
```

Base Handler

```
component{

    function aroundHandler(event,targetAction){
        var prc = event.getCollection(private=true);
        log.debug("Request for route: #event.getCurrentRoute()#",getHTTPRequestData());
        prc.results = prepareResults();

        arguments.targetAction(event,rc,prc);

        event.renderData(data=prc.results,type="json");
    }
    private function prepareResults(){
        var results = {
            error      = false,
            messages   = "",
            data       = ""
        };
        return results;
    }
    function onError(event,faultAction,exception){
        var rc = event.getCollection();
        rc.results.error   = true;
        rc.results.messages = "The action:#faultaction# failed:
                               #exception.detail# #exception.message#";

        log.error(rc.results.messages, exception);

        // Headers
        event.setHTTPHeader(500,rc.results.messages)
           .renderData(data=rc.results,type="json");
    }
}
```

Base Handler

```
component{

    function aroundHandler(event,targetAction){
        var prc = event.getCollection(private=true);
        log.debug("Request for route: #event.getCurrentRoute()#",getHTTPRequestData());
        prc.results = prepareResults();

        arguments.targetAction(event,rc,prc);

        event.renderData(data=prc.results,type="json");
    }
    private function prepareResults(){
        var results = {
            error      = false,
            messages   = "",
            data       = ""
        };
        return results;
    }
    function onError(event,faultAction,exception){
        var rc = event.getCollection();
        rc.results.error   = true;
        rc.results.messages = "The action:#faultaction# failed:
                               #exception.detail# #exception.message#";

        log.error(rc.results.messages, exception);

        // Headers
        event.setHTTPHeader(500,rc.results.messages)
           .renderData(data=rc.results,type="json");
    }
}
```

Base Handler

```
component{

    function aroundHandler(event,targetAction){
        var prc = event.getCollection(private=true);
        log.debug("Request for route: #event.getCurrentRoute()#",getHTTPRequestData());
        prc.results = prepareResults();

        arguments.targetAction(event,rc,prc);

        event.renderData(data=prc.results,type="json");
    }
    private function prepareResults(){
        var results = {
            error      = false,
            messages   = "",
            data       = ""
        };
        return results;
    }
    function onError(event,faultAction,exception){
        var rc = event.getCollection();
        rc.results.error   = true;
        rc.results.messages = "The action:#faultaction# failed:
                               #exception.detail# #exception.message#";

        log.error(rc.results.messages, exception);

        // Headers
        event.setHTTPHeader(500,rc.results.messages)
           .renderData(data=rc.results,type="json");
    }
}
```

Base Handler

```
component{

    function aroundHandler(event,targetAction){
        var prc = event.getCollection(private=true);
        log.debug("Request for route: #event.getCurrentRoute()#",getHTTPRequestData());
        prc.results = prepareResults();

        arguments.targetAction(event,rc,prc);

        event.renderData(data=prc.results,type="json");
    }
    private function prepareResults(){
        var results = {
            error      = false,
            messages   = "",
            data       = ""
        };
        return results;
    }
    function onError(event,faultAction,exception){
        var rc = event.getCollection();
        rc.results.error   = true;
        rc.results.messages = "The action:#faultaction# failed:
                               #exception.detail# #exception.message#";

        log.error(rc.results.messages, exception);

        // Headers
        event.setHTTPHeader(500,rc.results.messages)
           .renderData(data=rc.results,type="json");
    }
}
```

Base Handler

```
component{

    function aroundHandler(event,targetAction){
        var prc = event.getCollection(private=true);
        log.debug("Request for route: #event.getCurrentRoute()#",getHTTPRequestData());
        prc.results = prepareResults();

        arguments.targetAction(event,rc,prc);

        event.renderData(data=prc.results,type="json");
    }
    private function prepareResults(){
        var results = {
            error      = false,
            messages   = "",
            data       = ""
        };
        return results;
    }
    function onError(event,faultAction,exception){
        var rc = event.getCollection();
        rc.results.error   = true;
        rc.results.messages = "The action:#faultaction# failed:
                               #exception.detail# #exception.message#";

        log.error(rc.results.messages, exception);

        // Headers
        event.setHTTPHeader(500,rc.results.messages)
           .renderData(data=rc.results,type="json");
    }
}
```

Base Handler

```
component{

    function aroundHandler(event,targetAction){
        var prc = event.getCollection(private=true);
        log.debug("Request for route: #event.getCurrentRoute()#",getHTTPRequestData());
        prc.results = prepareResults();

        arguments.targetAction(event,rc,prc);

        event.renderData(data=prc.results,type="json");
    }
    private function prepareResults(){
        var results = {
            error      = false,
            messages   = "",
            data       = ""
        };
        return results;
    }
    function onError(event,faultAction,exception){
        var rc = event.getCollection();
        rc.results.error   = true;
        rc.results.messages = "The action:#faultaction# failed:
                               #exception.detail# #exception.message#";

        log.error(rc.results.messages, exception);

        // Headers
        event.setHTTPHeader(500,rc.results.messages)
           .renderData(data=rc.results,type="json");
    }
}
```

RESTful Event Handlers

RESTful Event Handlers

```
component cacheTimeout="30"{
    // DI
    property name="userService" inject;
    // HTTP Security
    this.allowedMethods = {
        index = "get,post", echo="post,put", remove="delete"
    };

    function index(event,rc,prc){
        prc.results.data = userService.get(id=event.getValue("id",0));
    }

    function echo(event,rc,prc){
        prc.results.data = "My name is #event.getValue("name","nobody")#");
    }
}
```

- Handler Persistence
- Autowire Domain Objects
- HTTP Action Security
- AOP Transformations

RESTful Event Handlers

```
component cacheTimeout="30"{
    // DI
    property name="userService" inject;
    // HTTP Security
    this.allowedMethods = {
        index = "get,post", echo="post,put", remove="delete"
    };

    function index(event,rc,prc){
        prc.results.data = userService.get(id=event.getValue("id",0));
    }

    function echo(event,rc,prc){
        prc.results.data = "My name is #event.getValue("name","nobody")#");
    }
}
```

- Handler Persistence
- Autowire Domain Objects
- HTTP Action Security
- AOP Transformations

RESTful Event Handlers

```
component cacheTimeout="30"{
    // DI
    property name="userService" inject;
    // HTTP Security
    this.allowedMethods = {
        index = "get,post", echo="post,put", remove="delete"
    };

    function index(event,rc,prc){
        prc.results.data = userService.get(id=event.getValue("id",0));
    }

    function echo(event,rc,prc){
        prc.results.data = "My name is #event.getValue("name","nobody")#");
    }
}
```

- Handler Persistence
- Autowire Domain Objects
- HTTP Action Security
- AOP Transformations

RESTful Event Handlers

RESTful Event Handlers

- Create two resources:
- /api/data
  GET
- /api/echo/:nickname
  GET then POST

RESTful Results

POST http://app.com/api/echo {name=”Louis Mahoney”}
StatusCode = 200

GET http://app.com/api/echo/hacker
StatusCode = 403

RESTful Results

POST http://app.com/api/echo {name=”Louis Mahoney”}
StatusCode = 200

```
{error:'false', messages : '', data: 'My Name is Louis Mahoney'}

{error:'false', messages : '', data: 'My Name is Josue TierraAlta'}
```

GET http://app.com/api/echo/hacker
StatusCode = 403

```
{error:'true', messages : 'HTTP Verb Security Action Echo cannot be executed via a
GET', data: ''}
```

Integration Testing

Integration Testing

- Top-Down test of your entire app
- Target specific events
- Assert transformations and data retrievals
- Code confidence
- HTTP Verb Mocking
- Data Mocking

Integration Testing

Integration Testing

```
component extends="coldbox.system.testing.BaseTestCase"{


    function testEchoNoName(){
        event = execute("general.echo");
        assertEquals();
    }

    function testEchoWithName(){
        URL.name = "Louis Mahoney";
        event = execute("general.echo");
        assertEquals();
    }

    function testEchoWithInvalidVerb(){
        mockContext = getMockBox()
                        .prepareMock( getRequestContext() );
        mockContext.$("getHTTPVerb","GET");
        event = execute("general.echo");
        assertEquals();
    }

}
```

Integration Testing

```
component extends="coldbox.system.testing.BaseTestCase"{

    function testEchoNoName(){
        event = execute("general.echo");
        assertEquals();
    }

    function testEchoWithName(){
        URL.name = "Louis Mahoney";
        event = execute("general.echo");
        assertEquals();
    }

    function testEchoWithInvalidVerb(){
        mockContext = getMockBox()
                        .prepareMock( getRequestContext() );
        mockContext.$("getHTTPVerb","GET");
        event = execute("general.echo");
        assertEquals();
    }

}
```

Integration Testing

```
component extends="coldbox.system.testing.BaseTestCase"{

    function testEchoNoName(){
        event = execute("general.echo");
        assertEquals();
    }

    function testEchoWithName(){
        URL.name = "Louis Mahoney";
        event = execute("general.echo");
        assertEquals();
    }

    function testEchoWithInvalidVerb(){
        mockContext = getMockBox()
                        .prepareMock( getRequestContext() );
        mockContext.$("getHTTPVerb","GET");
        event = execute("general.echo");
        assertEquals();
    }

}
```

Integration Testing

```
component extends="coldbox.system.testing.BaseTestCase"{

    function testEchoNoName(){
        event = execute("general.echo");
        assertEquals();
    }

    function testEchoWithName(){
        URL.name = "Louis Mahoney";
        event = execute("general.echo");
        assertEquals();
    }

    function testEchoWithInvalidVerb(){
        mockContext = getMockBox()
                        .prepareMock( getRequestContext() );
        mockContext.$("getHTTPVerb","GET");
        event = execute("general.echo");
        assertEquals();
    }

}
```

ColdBox Resources

- Official Site
  - www.coldbox.org
- Documentation
  - wiki.coldbox.org
- Google Group
  - groups.google.com/group/coldbox
- Training
  - www.coldbox.org/training
- Professional Support
  - www.ortussolutions.com

ColdBox Resources

- Official Site
  - www.coldbox.org
- Documentation
  - wiki.coldbox.org
- Google Group
  - groups.google.com/group/coldbox
- Training
  - www.coldbox.org/training
- Professional Support
  - www.ortussolutions.com

Luis Majano &
Ortus Solutions, Corp
lmajano@ortussolutions.com

ColdBox Resources

- Official Site
  - www.coldbox.org
- Documentation
  - wiki.coldbox.org
- Google Group
  - groups.google.com/group/coldbox
- Training
  - www.coldbox.org/training
- Professional Support
  - www.ortussolutions.com

Luis Majano &
Ortus Solutions, Corp
lmajano@ortussolutions.com

Q & A

Thanks!

Q & A

ColdBox

Thanks!