ERRest in Depth

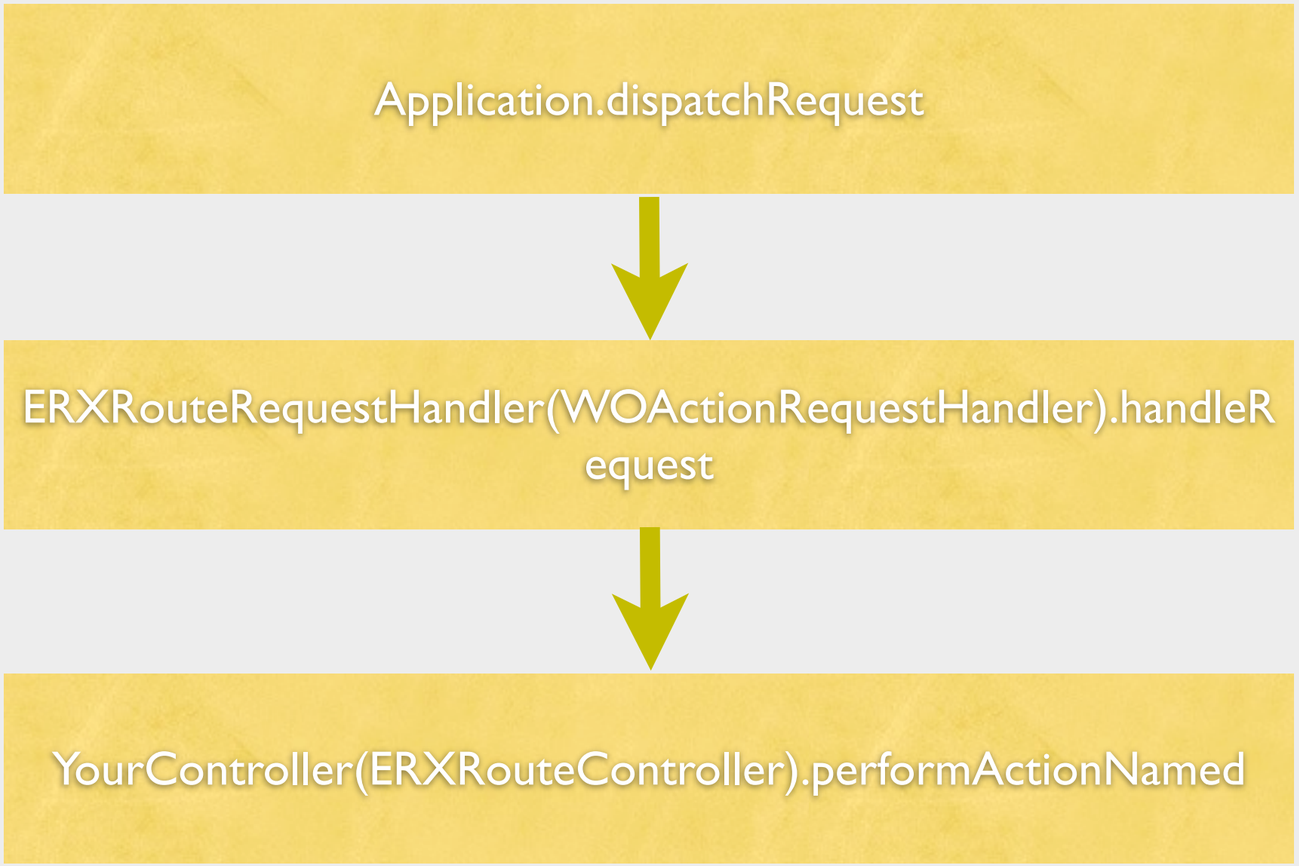
Pascal Robert
MacTI.ca

The Menu

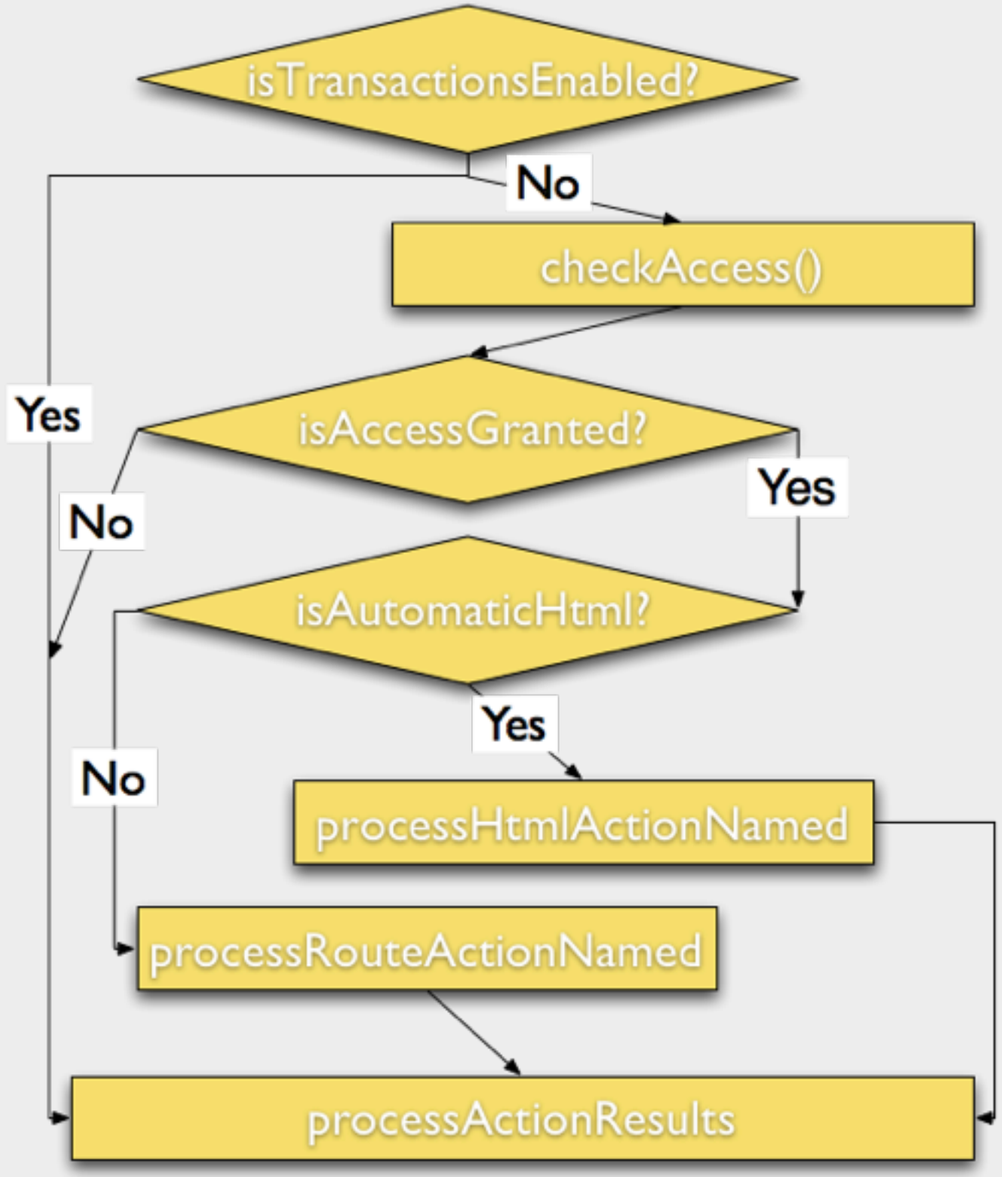
- Request/response loop
- Behavior changes
- Formats
- Transactions
- Same Origin Policy
- Date and time management

Request handling

Request


Application.dispatchRequest
ERXRouteRequestHandler(WOActionRequestHandler).handleRequest
YourController(ERXRouteController).performActionNamed

performActionNamed



checkAccess()

- Default implementation in ERXRouteController does nothing
- Override it in your controller for security check

performHtmlActionNamed

- Does <EntityName><ActionName>Page component exists?
  - No: fall back to controller
  - Yes: Check if component implements IEXRouteComponent
    - Yes: return the component
    - No: fall back to controller

shouldFailOnMissingHtmlPage

- Does the component was not found or don't implement IEXRouteComponent?
  - If shouldFailOnMissingHtmlPage() returns true, call performUnknownAction (will return a 404 NotFound)
  - Default is false, override it in your controller if needed.

performRouteActionNamed

- Try to find <actionName>Action method.
  - Not found? Try to find <actionName> method.
  - Still nothing? Check for annotations.
  - Still nothing? Call performUnknownAction
  - Got something? Call performActionWithArguments

performUnknownAction

- if (ERXRest.strictMode)
  - throw ERXNotAllowedException (HTTP code 405)
- else
  - throw FileNotFoundException (HTTP code 404)

performActionWithArguments

- Will invoke the method with the arguments

Objects management

Objects in routes

- /ra/<entityName>/{entity:EntityName}
- routeObjectForKey(key)
- create(filter)
- update(object, filter)

routeObjectForKey

- {<keyName>:<keyType>} in route = keyType result = routeObjectForKey(<keyName>)
- Object is obtained by ERXRestUtils.coerceValueToTypeNamed

coerceValueToTypeNamed

- Where value is transformed to a object or primitive
- If it's an EO or POJO, will use ERXRestClassDescriptionFactory.classDescriptionForEntityName to find the class
- Will call IERXRestDelegate.Factory.delegateForClassDescription().objectOfEntityWithID()

create(filter)

- Will call ERXRestClassDescriptionFactory.classDescriptionForEntityName
- Will call IERXRestDelegate.Factory.delegateForClassDescription().createObjectOfEntityWithID to create a basic EO/POJO
- Will call updateObjectWithFilter()

update(object, filter)

- As with create(filter), will simply call updateObjectWithFilter

updateObjectWithFilter

- Major method
- Will take content from request and update object (PUT request)
- Also used to populate new object (POST request)

Response handling

response(object, filter)

- Will built up the object graph with ERXRestRequestNode.requestNodeWithObjectAndFilter
- Will call response(format, responseNode)

requestNodeWithObjectAndFilter

- If primitive/simple object, will set the value
- If EO/POJO, will call _fillInWithObjectAndFilter

_fillInWithObjectAndFilter

- If object is an array, method take itself
- If not array, will use object's delegate to call primaryKeyForObject, and call _addAttributesAndRelationshipsForObjectOfEntity

response(format, responseNode)

- Returns result of ERXRouteResults(context, restContext, format, responseNode)
- ERXRouteResults.generateResponse() will actually generate the response in requested format

response(ERXRestFetchSpecification, filter)

- Useful to return list of objects ("index" action)
- ERXRestFetchSpecification allow you to set ordering, range, filtering and batching from request
- Will also call response(format, responseNode)

response(int)

- Use that one to send a response without any content in the body
- Check ERXHttpStatusCodes

There's a property for that

"id" key

- ERXRest.idKey : what to use instead of "id"

  Default: {"id":2 }

  ERRest.idKey=primaryKey {"primaryKey":2 }

"nil" key

- ERXRest.nilKey

  To rewrite the "nil" attribute

  Default: <someAttribute nil="true"/>

  ERXRest.nilKey=cestVide -> <someAttribute cestVide="true"/>

- ERXRest.writeNilKey

  Skip the "nil" attribute

  Default: <someAttribute nil="true"/>

  ERXRest.writeNilKey=false -> <someAttribute ~~nil="true"~~/>

"type" key

- ERXRest.typeKey

  Allow you to change the name of the "type" attribute

  Default: {"type":"NameOfEntity"}

  ERXRest.typeKey=entityName {"entityName":"NameOfEntity"}

- ERXRest.writeTypeKey

  If false, won't write the "type" attribute in the response

  ERXRest.writeTypeKey=false -> {id: 2, ~~"type":"NameOfEntity"~~}

ERXRest.pluralEntityNames

- Default is true
- If set to false, can't use pluralized names

  Default: /ra/restEntities

  Set to false: ~~/ra/restEntities~~

ERXRest.suppressTypeAttributesForSimpleTypes

- Only for XML format
- Default value is false
- Default rendering:

```
<RestEntity primaryKey="1">
  <someAttribute type = "integer">2</someAttribute>
</RestEntity>
```

- When set to false:

```
<RestEntity primaryKey="1">
  <someAttribute ~~type = "integer"~~>2</someAttribute>
```

ERXRest.strictMode

- Default is: true
- For missing route, will send 405 (Not Allowed) code, if set to false, will send 404 (Not Found)
- POST requests: will send 201 (Created), if false will send 200 (OK)

ERXRest.routeCase

- ERXRest.routeCase=LowerCamelCase

  /ra/restEntities
- ERXRest.routeCase=CamelCase

  /ra/RestEntities
- ERXRest.routeCase=LowercaseUnderscore

  /ra/rest_entities

ERXRest.parseUnknownExtensions

- /ra/restEntities/3.fsdfsd
- If set to true (the default):

  HTTP/1.1 200 Apple WebObjects

  Content-Type: text/html
- If set to false:

  HTTP/1.0 400 Apple WebObjects

Formats

ERXRest.defaultFormat

- ERXRest.defaultFormat=json|xml|plist|html|...
- Let you specify the default format for all controllers
- Can override it per controller:

```
protected ERXRestFormat defaultFormat() {
  return ERXRestFormat.json();
}
```

Format detection

- Format detection is in ERXRouteController.format()
- Order of detection
  - From extension (.json). Set in ERXRouteRequestHandler.routeForMethodAndPath()
  - From the Content-Type header
  - Default format (defaultFormat() in controller)

Adding new format

- Your own private format? Use ERXRestFormat.registerFormatNamed()
- Format useful for the commumity? Add them to ERXRestFormat

Same Domain Policy

Same Origin Policy

- Problem: browsers won't load data if client and server not on same domain
- Numerous ways: window.name transport, JSONP and Cross-origin resource sharing (*CORS*)
- CORS and window.name transport support is part of ERRest

CORS

- Works with Gecko 1.9.1 (Firefox 3.5+), WebKit (Safari 4+, Google Chrome 3+), Opera 12 and IE 8+
- Send extra headers to server
- Client specifiy origin, requested HTTP verb and allowed headers, server returns allowed origin, methods, headers and max-age

CORS preflight

Client request:

```
OPTIONS /resources/post-here/ HTTP/1.1
Origin: http://foo.example
Access-Control-Request-Method: POST
Access-Control-Request-Headers: X-PINGOTHER
```

Server response:

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: http://foo.example
Access-Control-Allow-Methods: POST, GET, OPTIONS
Access-Control-Allow-Headers: X-PINGOTHER
Access-Control-Max-Age: 1728000
```

Client request:

```
GET /resources/post-here/ HTTP/1.1
Origin: http://foo.example
```

Server response:

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: http://foo.example
```

CORS

- ERXRest.accessControlAllowRequestHeaders
- ERXRest.accessControlAllowRequestMethods
- ERXRest.accessControlMaxAge (default: 1728000)
- ERXRest.accessControlAllowOrigin ('*' to allow all)

window.name Transport

- Dojo use that trick
- Client send ?windowname=true in URL
- Enable it on server with:

  ERXRest.allowWindowNameCrossDomainTransport=true

- Will wrap response in HTML code:

  <html><script type="text/javascript">window.name='{"id":
  4,"type":"RestEntity","someAttribute":"commit transaction"}';</script></html>

Status codes

Status code and exceptions

- ObjectNotAvailableException, FileNotFoundException, NoSuchElementException: returns a 404 (Not Found)
- SecurityException: returns a 403 (Forbidden)
- ERXNotAllowedException: returns a 405 (Method Not Allowed)
- ERXValidationException, NSValidation.ValidationException: returns a 400 (Bad Request)
- Anything else: returns a 500 (Internal Server Error)
- Avoid sending 5xx codes if the client made a mistake!

Adding support for new data types

- Add a processor in _ERXJSONConfig
- Add support code in ERXRestUtils
  - isPrimitive()
  - coerceValueToString()
  - coerceValueToTypeNamed
- For dates: add formatter in ERXRestUtils

Q&A